The background of the slide features a complex, abstract simulation of a fluid or plasma. The colors are primarily shades of blue, green, and yellow, with intricate patterns of swirling and mixing. The overall texture is organic and dynamic, suggesting a three-dimensional flow.

Università di Bologna

# Tangent-based manifold approximation with locally linear models

**Adriano Donninelli**

un lavoro di Sofia Karygianni, Pascal Frossard

Partendo dal fatto che dati appartenenti a spazi dimensionalmente complessi siano difficili da trattare (the curse of dimensionality) si osserva che spesso essi possiedono strutture sottostanti che possono permettere una rappresentazione adeguata in dimensione inferiore.

In questo lavoro ci si propone di costruire un algoritmo in grado di approssimare un set di dati attraverso un modello semplice e computazionalmente efficiente ossia dei sottospazi affini di dimensione fissata cercando di preservare globalmente la geometria della struttura di partenza.

In generale le varietà (Manifold) sono strutture globalmente complesse che localmente ossia vicino ad ogni loro punto possiedono le stesse caratteristiche dello spazio Euclideo. In questo lavoro, si considerano varietà  $d$ -dimensionali (differenziabili) immerse in uno spazio Euclideo di dimensione  $\mathbb{R}^N$  con  $N \gg d$ . Intuitivamente si può pensare a una varietà  $d$ -dimensionale immersa in  $\mathbb{R}^N$  come alla generalizzazione di una superficie in  $N$  dimensioni: un insieme di punti che localmente sembra vivere in  $\mathbb{R}^d$  ma che macroscopicamente sintetizza una struttura in  $\mathbb{R}^N$ .

Un esempio classico sono una sfera in  $\mathbb{R}^3$  e una circonferenza in  $\mathbb{R}^2$ , rispettivamente due varietà di dimensione 2 e 1.

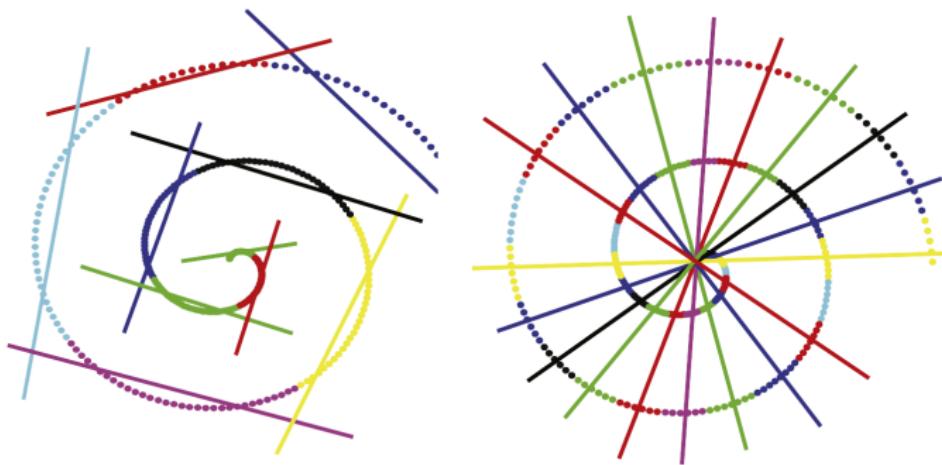


Figura: A sinistra una buona approssimazione di una varietà 1D immersa in  $\mathbb{R}^2$  dove i colori rappresentano i diversi gruppi di punti approssimati da linee. A destra una cattiva approssimazione che non preserva la geometria della varietà.

## Definizione (Varietà)

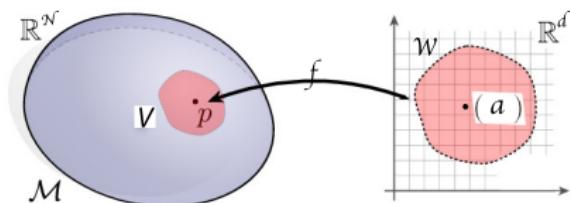
Un insieme  $M \subseteq \mathbb{R}^N$  è detto varietà differenziabile di dimensione  $d$  se  $\forall x \in M$  esiste un aperto  $V \subseteq \mathbb{R}^N$  t.c.  $x \in V$  e un aperto  $W \subseteq \mathbb{R}^d$  ed esiste  $f : W \rightarrow \mathbb{R}^N$  iniettiva, differenziabile e ad inversa continua t.c.

- $f(W) = M \cap V$  e
- $\text{Rank}(Df(y)) = d \quad \forall y \in W$  (Dove  $Df$  indica la Jacobiana di  $f$ )

Supponiamo  $f(a) = x$  allora la matrice  $Df(a)$  e la corrispondente trasformazione lineare  $f_* : \mathbb{R}_a^d \rightarrow \mathbb{R}_x^N$  definiscono un sottospazio di dimensione  $d$  detto lo spazio tangente a  $M$  in  $x$  denotato  $M_x$ .

Per comodità indicheremo da qui in poi con  $M_x$  lo spazio tangente ad  $M$  in  $x$  traslato però all'origine di  $\mathbb{R}^N$ .

## Osservazione motivante

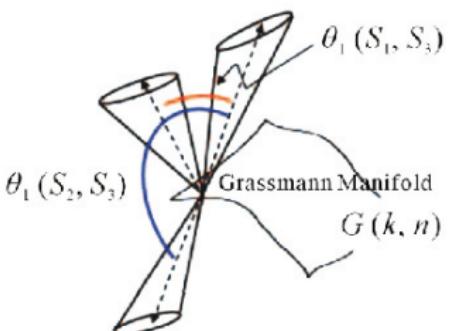


È fondamentale osservare che se per opportuni  $x, V$  e  $W$   $f$  è lineare allora  $Df(a) = Df(b) \forall a, b \in W$  e dunque lo spazio tangente di tutti i punti in  $M \cap V$  coincide (a meno di una traslazione) e la regione di spazio può essere perfettamente rappresentata con dei sottospazi affini. Siamo allora interessati a determinare regioni dove la *variazione* dello spazio tangente nei diversi punti è bassa (così  $Df(a) \approx Df(b)$ ). Definiamo dunque una metrica tra spazi tangenti.

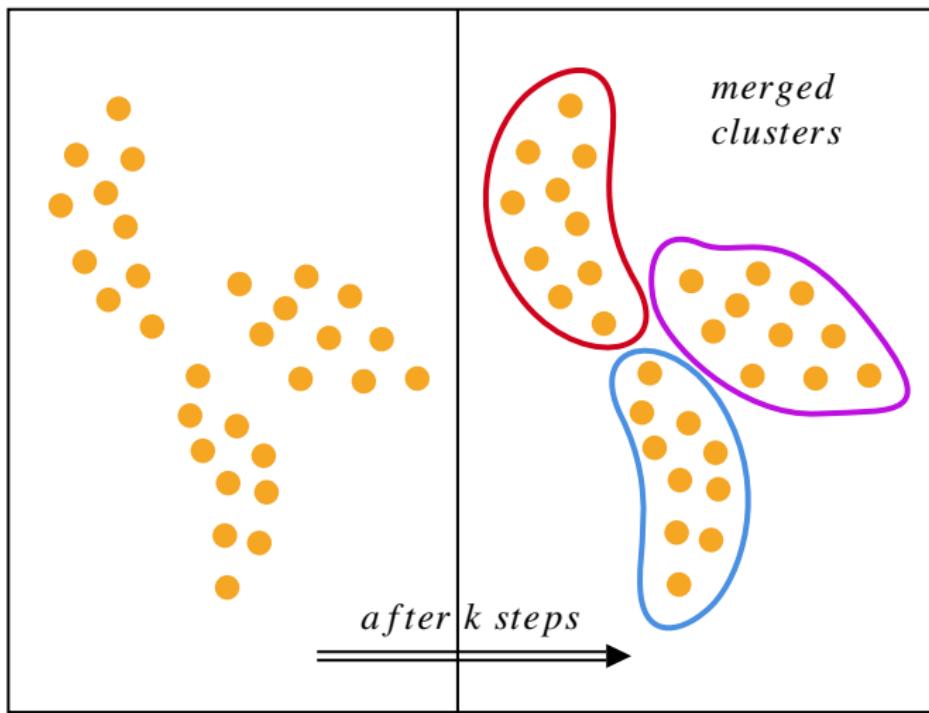
L'insieme dei sottospazi lineari di dimensione  $d$  di  $\mathbb{R}^N$  è detto Grassmanniana indicato  $G_{N,d}(\mathbb{R}^N)$ . In  $G_{N,d}(\mathbb{R}^N)$  è definita la distanza geodetica tra due sottospazi a partire dai loro angoli principali. In particolare possiamo definire la distanza tra  $M_x$  e  $M_y$  come

$$D_T(M_x, M_y) = \left( \sum_{i=1}^d \theta_i^2 \right)^{\frac{1}{2}} = \|\theta\|_2$$

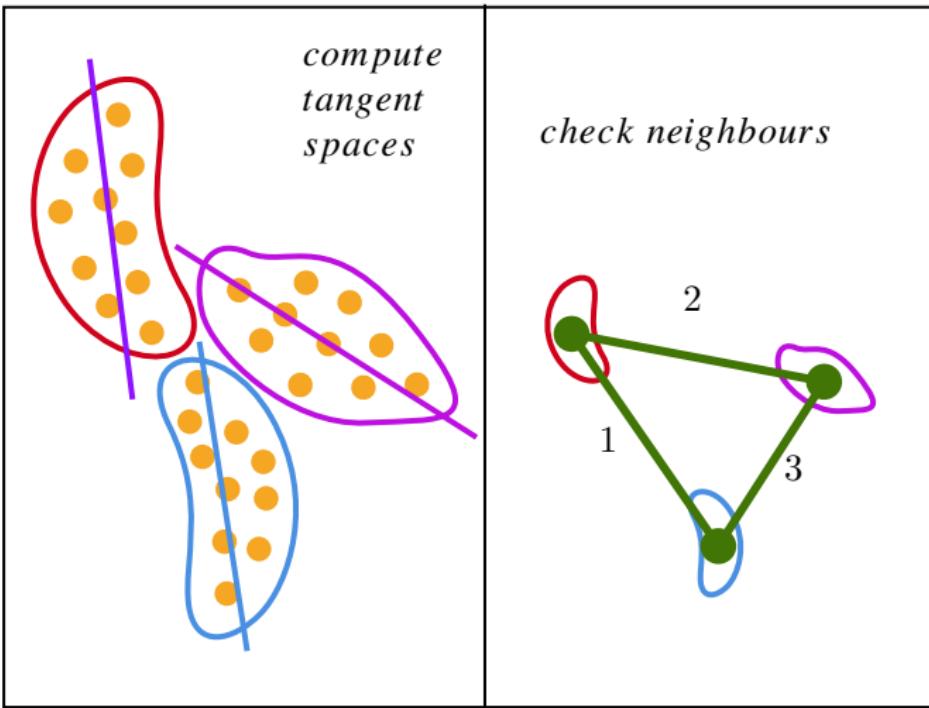
dove  $\theta = \{\theta_1, \dots, \theta_d\}$  è il vettore degli angoli principali di  $M_x$  e  $M_y$ . Computabili attraverso SVD come mostrato da Andrew V. Knyazev et al. (2006)



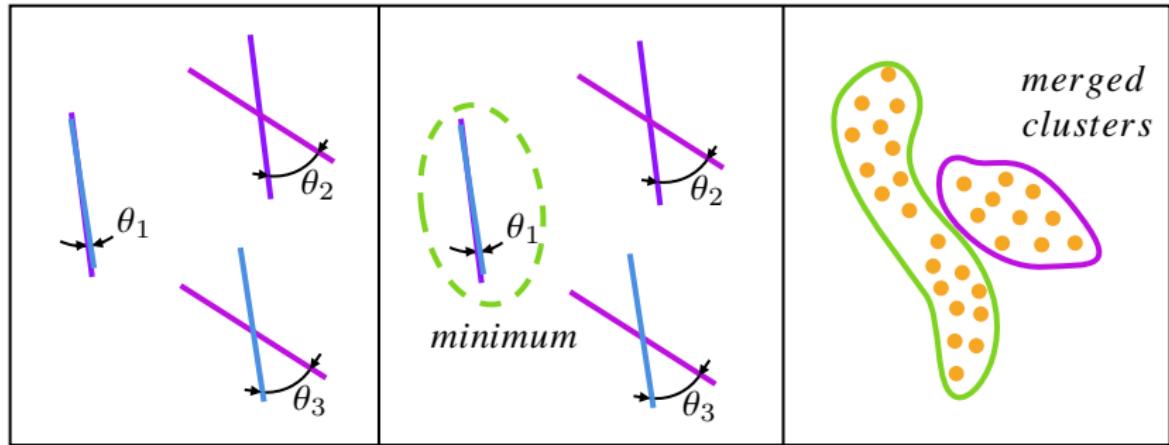
Intuitivamente, in  $\mathbb{R}^2$ ,  $d = 1$



Intuitivamente, in  $\mathbb{R}^2$ ,  $d = 1$



Intuitivamente, in  $\mathbb{R}^2$ ,  $d = 1$



Siamo inoltre interessati a poter calcolare una generalizzazione della media aritmetica applicata alla Grassmanniana.

**Definizione (Punto medio di Karcher su  $G_{N,d}(\mathbb{R}^N)$  rispetto  $D_T$ )**

Il punto medio di un insieme  $C$  di punti di  $G_{N,d}(\mathbb{R}^N)$  rispetto alla distanza  $D_T$  è dato da

$$M_C = \arg \min_{M \in G_{N,d}} \sum_{x \in C} D_T^2(M_x, M)$$

Esistono vari metodi per risolvere in  $M_C$  la precedente equazione, qui viene usata una formulazione data da *J.M.Chang* che sfrutta la decomposizione in valori singolari (SVD).

Sia  $\mathcal{X} = \{x_k \in \mathbb{R}^N, k \in [1, m]\}$  la rappresentazione di una varietà attraverso un insieme di suoi punti. Sia  $G_{\mathcal{X}} = G(\mathcal{X}, E)$  il grafo simmetrico non orientato che rappresenta la geometria della varietà. Vogliamo determinare  $\mathbf{C}_{\mathcal{L}} = \{C_i, i \in [1, \mathcal{L}]\}$  partizione di  $\mathcal{X}$  tale che  $\forall i$   $C_i$  possa essere ben rappresentato da un sottospazio affine che rispetti la geometria della varietà.

## Definizione (Partizione)

$\mathbf{C}_{\mathcal{L}}$  è una partizione se

$$C_i \cap C_j = \emptyset \quad \forall i \neq j, \quad i, j \in \{1, \dots, \mathcal{L}\} \quad \text{e} \quad \bigcup_i C_i = \mathcal{X}.$$

Non tutte le partizioni saranno però valide, una condizione sufficiente è che la partizione sia formata solo da cluster con sottografi connessi.

## Definizione (Sottografo connesso)

Dato  $G_{C_i} = G(C_i, E_i)$  sottografo dove  $E_i = \{a_{ij} \in E : x_i, x_j \in C_i\}$  diciamo che esso è connesso se ogni coppia di nodi  $x_i, x_j \in C_i$  è connessa.

Denotiamo l'insieme delle partizioni valide  $\Phi_{\mathcal{L}}(\mathcal{X})$ .

## Definizione (Predicato di Validità)

$\Phi_{\mathcal{X}}(\mathbf{C}_{\mathcal{L}}) \equiv \mathbf{C}_{\mathcal{L}} \in \Phi_{\mathcal{L}}(\mathcal{X})$  definito come

$$\Phi_{\mathcal{X}}(\mathbf{C}_{\mathcal{L}}) = \bigwedge_{C_i \in \mathbf{C}_{\mathcal{L}}} \phi(C_i)$$

$$\phi(C_i) = \begin{cases} \text{Vero} & \text{Se } C_i \text{ è connesso} \\ \text{Falso} & \text{altrimenti} \end{cases}$$

Possiamo definire anche un predicato di fondibilità  $\Psi$  che descrive se due cluster possono essere uniti dando vita a una partizione ancora valida

## Definizione (Predicato di Fondibilità (informale))

I predicati di fondibilità e validità sono legati dalla relazione:  
Se  $C_i, C_j \neq \emptyset, C_i \cap C_j = \emptyset, \phi(C_i) \wedge \phi(C_j)$  e  $\Psi(C_i, C_j) \implies \phi(C_i \cup C_j)$

Evitando una definizione formale, due cluster saranno fondibili se esiste un *edge* che li collega.

## Cluster fondibili e non fondibili

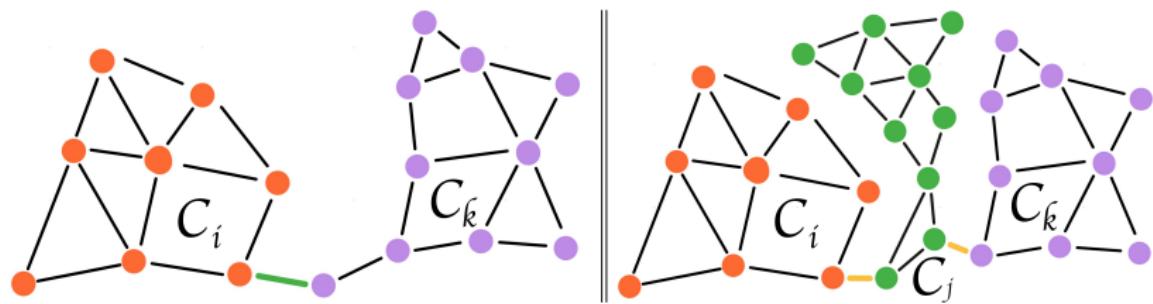


Figura: A sinistra due cluster  $C_i$  e  $C_k$  connessi e dunque fondibili. A destra tre cluster dove sia  $C_i$  che  $C_k$  sono fondibili con  $C_j$  ma  $C_i$  e  $C_k$  non sono fondibili. È facile osservare che fondendo le coppie ( valide ) in entrambe le configurazioni la partizione risultante è composta da sottografi connessi e dunque valida.

## Bontà di una partizione

Cerchiamo ora una funzione  $\mathcal{P}$  che valuti la bontà di una partizione, definiamo

$$\mathcal{P}(\mathbf{C}) = \sum_{C_i \in \mathbf{C}} p(C_i) \text{ con}$$

$$p(C_i) = \sum_{x \in C_i} D_T^2(M_{C_i}, M_x)$$

Osserviamo che per definizione  $\mathcal{P}$  è distributiva sui cluster e non negativa, inoltre è nulla sui cluster formati da singoletti. Ora che abbiamo una misura di bontà possiamo definire il nostro problema di approssimazione come la ricerca della partizione  $\mathbf{C}_{\mathcal{L}}^*(X)$  dove

$$\mathbf{C}_{\mathcal{L}}^*(X) = \operatorname{argmin}_{\mathbf{C} \in \Phi_{\mathcal{L}}(X)} \mathcal{P}(C) = \operatorname{argmin}_{\mathbf{C} \in \Phi_{\mathcal{L}}(X)} \sum_{C_i \in \mathbf{C}} \sum_{x \in C_i} D_T^2(M_{C_i}, M_x)$$

# Verso una formulazione greedy

Vogliamo formulare una versione meno accurata ma più efficiente, introduciamo alcuni rilassamenti andando a determinare un algoritmo greedy e bottom-up, ci serve però un modo per valutare la bontà di una fusione di cluster.

## Definizione (Misura di Dissimilarità)

$$d : (C_i, C_j) \rightarrow R_0^+$$
$$d(C_i, C_j) = p(C_i \cup C_j) - p(C_i) - p(C_j)$$

È facile osservare che  $d \geq 0$ . (Appendice (3))

Possiamo allora riscrivere il problema nella forma

$$C_{\mathcal{L}}^*(X) = \left( C'_{\mathcal{L}+1}(X) \setminus \{C'_i, C'_j\} \right) \cup \{C'_i \cup C'_j\} \text{ dove}$$

$$\left( C'_{\mathcal{L}+1}(X), C'_i, C'_j \right) = \operatorname*{argmin}_{\substack{C_i, C_j \in C, C \in \Phi_{\mathcal{L}+1} \\ \Psi(C_i, C_j) \text{ è vera}}} (\mathcal{P}(C) + d(C_i, C_j))$$

## Approccio dinamico e greedy relaxation

Questa formulazione permette un approccio dinamico, per avere la migliore partizione di  $\mathcal{L}$  cluster possiamo controllare tutte le partizioni di  $\mathcal{L} + 1$  cluster e unire i due cluster a minor costo.

Deriviamo un algoritmo greedy togliendo la ricerca su  $\Phi_{\mathcal{L}+1}$  e otteniamo

$$\widehat{\mathbf{C}}_{\mathcal{L}}(\mathcal{X}) = \left( \mathbf{C}_{\mathcal{L}+1}(\mathcal{X}) \setminus \{C'_i, C'_j\} \right) \cup \{C'_i \cup C'_j\}$$

$$\text{dove } (C'_i, C'_j) = \underset{\substack{C_i, C_j \in \widehat{\mathbf{C}}_{\mathcal{L}+1}(\mathcal{X}) \\ \Psi(C_i, C_j) \text{ è vera}}}{\operatorname{argmin}} d(C_i, C_j)$$

Questo riduce notevolmente il costo computazionale ma non assicura più l'ottimalità della soluzione trovata.

Siamo ora pronti a descrivere l'algoritmo di clustering.

## Inizializzazione: $G_X$ e $M_x$

$G_X$  è costruita connettendo ogni punto ai suoi vicini attraverso una  $knn$ , per ogni punto  $x \in X$  definiamo inoltre  $N_x = \{y \in X : (x, y) \in E\}$  l'insieme dei neighbors di  $x$ . Ora  $M_x$  può essere approssimato come il sottospazio di dimensione  $d$  che meglio approssima  $N_x^0$ .

Per il teorema di *Eckart - Young* questo corrisponde a determinare la  $d$ -rank SVD di  $[N_x^0]$ .

## Algoritmo di merging (1)

Fissiamo  $n = |\mathcal{X}|$  e la partizione iniziale ottimale

$$C_n^* = \{\{x\}; x \in \mathcal{X}\}$$

Ad ogni iterazione uniamo due cluster  $C_i$  e  $C_j$ , Il predicato di fondibilità precedentemente definito ci permette di determinare tutti i possibili  $C_i$  e  $C_j$  fondibili: essi saranno i cluster che sono connessi per almeno un *edge*. La dissimilarità tra due cluster era definita come

$$\begin{aligned} d(C_i, C_j) &= p(C_i \cup C_j) - p(C_i) - p(C_j) \\ &= \sum_{x \in C_i} \left[ D_T^2 \left( M_x, M_{C_i \cup C_j} \right) - D_T^2 \left( M_x, M_{C_i} \right) \right] \\ &\quad + \sum_{x \in C_j} \left[ D_T^2 \left( M_x, M_{C_i \cup C_j} \right) - D_T^2 \left( M_x, M_{C_j} \right) \right] \end{aligned}$$

## Algoritmo di merging (2)

Tale equazione è costosa da computare poichè richiede il calcolo di  $M_{C_i \cup C_j}$  per ogni ipotetica fusione. Vorremmo una formulazione che dipenda solo da valori già calcolati e.g. lo spazio tangente medio di ogni cluster e la distanza tra esso e gli spazi tangenti dei singoli punti del cluster.

Osserviamo che  $\sum_{x \in C_i} D_T^2(M_x, M_{C_i \cup C_j}) \leq \sum_{x \in C_i} D_T^2(M_x, M_{C_j})$ , provabile p.a. (Appendice (1))

Lo stesso vale naturalmente per  $C_j$ , inoltre per la disuguaglianza triangolare

$$D_T(M_x, M_{C_j}) \leq D_T(M_x, M_{C_i}) + D_T(M_{C_i}, M_{C_j}), \quad \forall x \in X$$
$$D_T(M_x, M_{C_i}) \leq D_T(M_x, M_{C_j}) + D_T(M_{C_i}, M_{C_j}), \quad \forall x \in X$$

Allora elevando al quadrato e sommando sui punti di  $C_i$  e  $C_j$  rispettivamente otteniamo la formula finale per la nostra dissimilarità approssimata  $\tilde{d}$  che dipende solo da informazioni pregresse

## Algoritmo di merging (3)

$$d(C_i, C_j) \leq \tilde{d}(C_i, C_j) = (|C_i| + |C_j|) D_T^2(M_{C_i}, M_{C_j}) \\ + 2D_T(M_{C_i}, M_{C_j}) \left[ \sum_{x \in C_i} D_T(M_x, M_{C_i}) + \sum_{x \in C_j} D_T(M_x, M_{C_j}) \right]$$

Calcoleremo così lo spazio tangente medio solo per il nuovo cluster creato.

# Schema dell'algoritmo completo

**Input:**  $\mathcal{X}, k, \mathcal{L}, d$

**Step 1** \*Preprocessing, Section 5.1\*

1: Construct  $G(\mathcal{X}, E)$  by connecting each element in  $\mathcal{X}$  with its  $k$ -nearest neighbors.

2: **for all**  $x \in \mathcal{X}$  **do**

3:      $N_x = \{y \in \mathcal{X}: (x, y) \in E\}$  \*Compute neighborhoods\*

4:      $[N_x^0] = USV^T$

where  $[N_x^0]$  is the data matrix formed by the elements in  $N_x$  shifted to the origin of  $\mathbb{R}^N$  and  $U, S, V$  are the results of its d-rank SVD.

5:      $M_x = U$  \*Compute tangent spaces\*

6: **end for**

**Step 2** \*Greedy computation of partition  $\mathbf{C}_{\mathcal{L}}^*$ \*

7:  $n = |\mathcal{X}|$ ,  $\lambda = 0$ ,  $\mathbf{C}_n^* = \{x: x \in \mathcal{X}\}$  \*Initialization\*

8: **for**  $\lambda < n - \mathcal{L}$  **do** \*Greedy merging, Section 5.2\*

9:      $(C'_i, C'_j) = \underset{\substack{C'_i, C'_j \in \mathbf{C}_{n-\lambda}^* \\ w(C'_i, C'_j) \text{ min}}}{\operatorname{argmin}} d(C_i, C_j)$  \*Eq. (23)\*

10:      $\mathbf{C}_{n-\lambda+1}^* = (\mathbf{C}_{n-\lambda}^* \setminus \{C'_i, C'_j\}) \cup \{C'_i \cup C'_j\}$

11:     Compute  $M_{C'_i \cup C'_j}$  \*Eq. (3)\*

12:      $\lambda = \lambda + 1$

13: **end for**

14: **for**  $C_i \in \mathbf{C}_{\mathcal{L}}^*$  **do** \*Compute the final flats  $F_i$ \*

15:      $[C_{m_i}^0] = USV^T$

where  $m_i$  is the sample mean of  $C_i$ ,  $[C_{m_i}^0]$  is the data matrix formed by the samples in  $C_i$  shifted by  $m_i$  and  $U, S, V$  are the results of its d-rank SVD.

16:      $F_i = U$

17: **end for**

**Output:**  $\mathbf{C}_{\mathcal{L}}^*, F$

Nelle linee 14 – 16 vengono computati i sottospazi affini finali.

## Risultati sperimentali

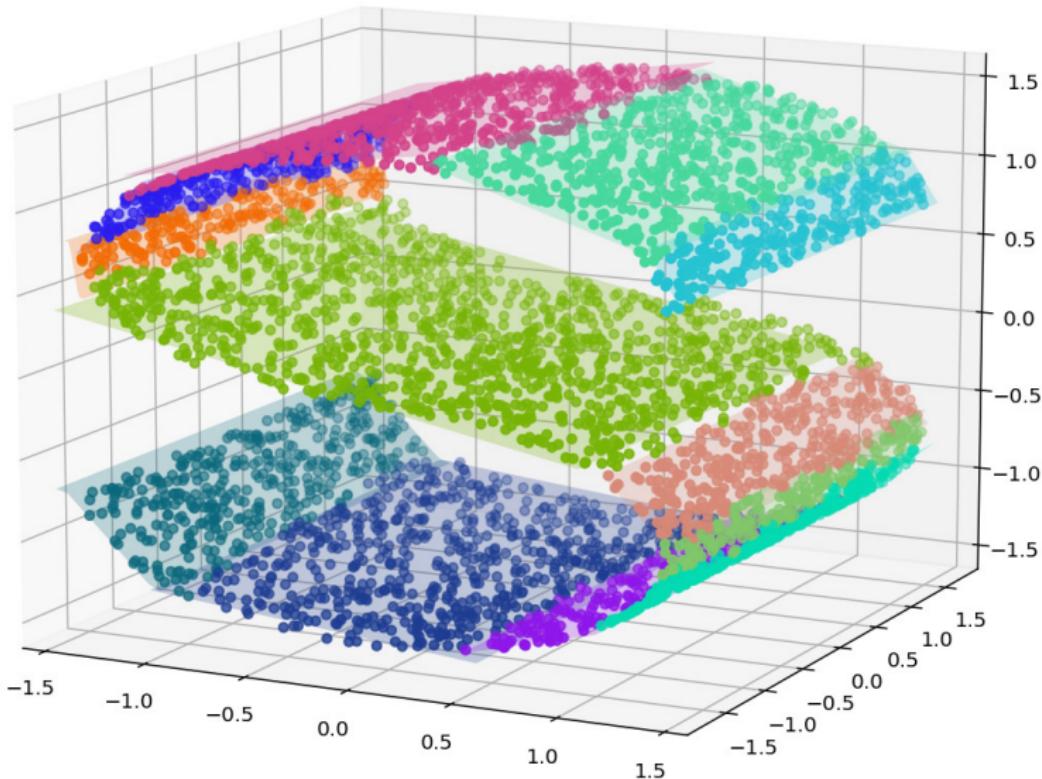
Iniziamo con l'approssimazione di 2 dataset sintetici 3D, la Swiss Roll e la S-Curve. I test sono condotti con 5000 punti, un neighborhood size di  $k = 15$  e  $d = 2$ , la performance è valutata con il *mean squared reconstruction error*.

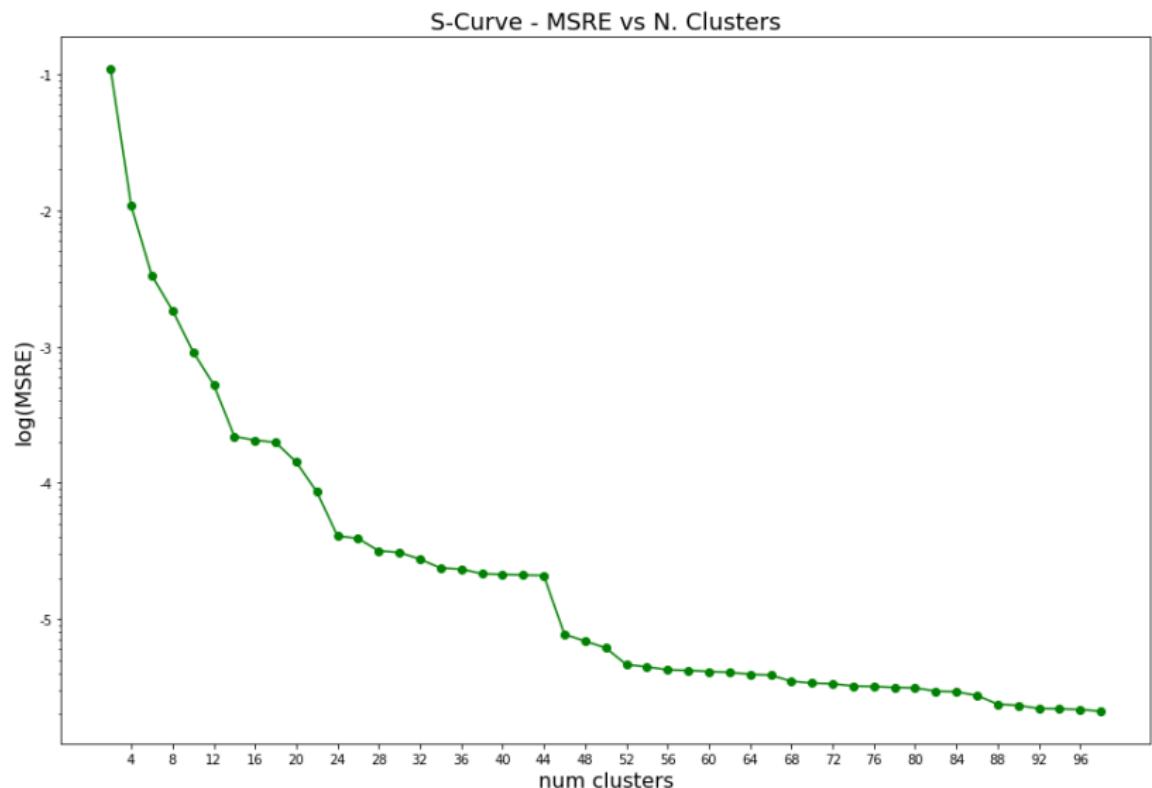
$$MSRE = \frac{1}{N} \sum_{i=1}^N \|x_i - \hat{x}_i\|^2.$$

Riportiamo anche i tempi di esecuzione dell'algoritmo su tali dataset:

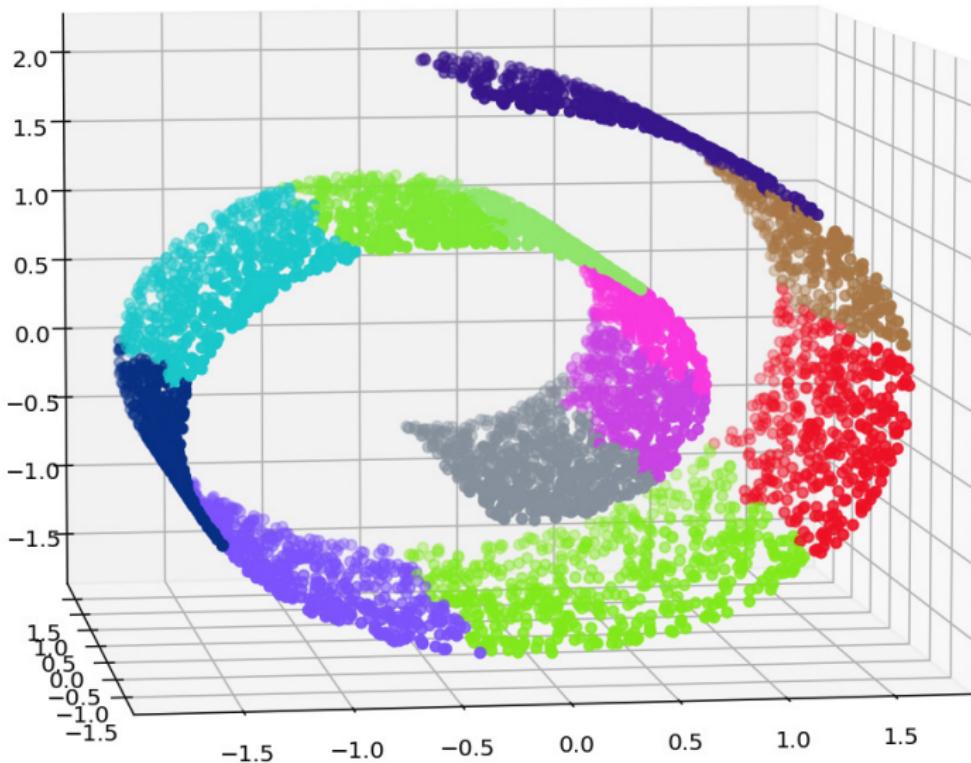
|                             |                           |                           |
|-----------------------------|---------------------------|---------------------------|
| Paper - MBP 2.66Ghz<br>389s | Mine - MBA 1.6Ghz<br>267s | Mine - i5 3.60GHz<br>123s |
|-----------------------------|---------------------------|---------------------------|

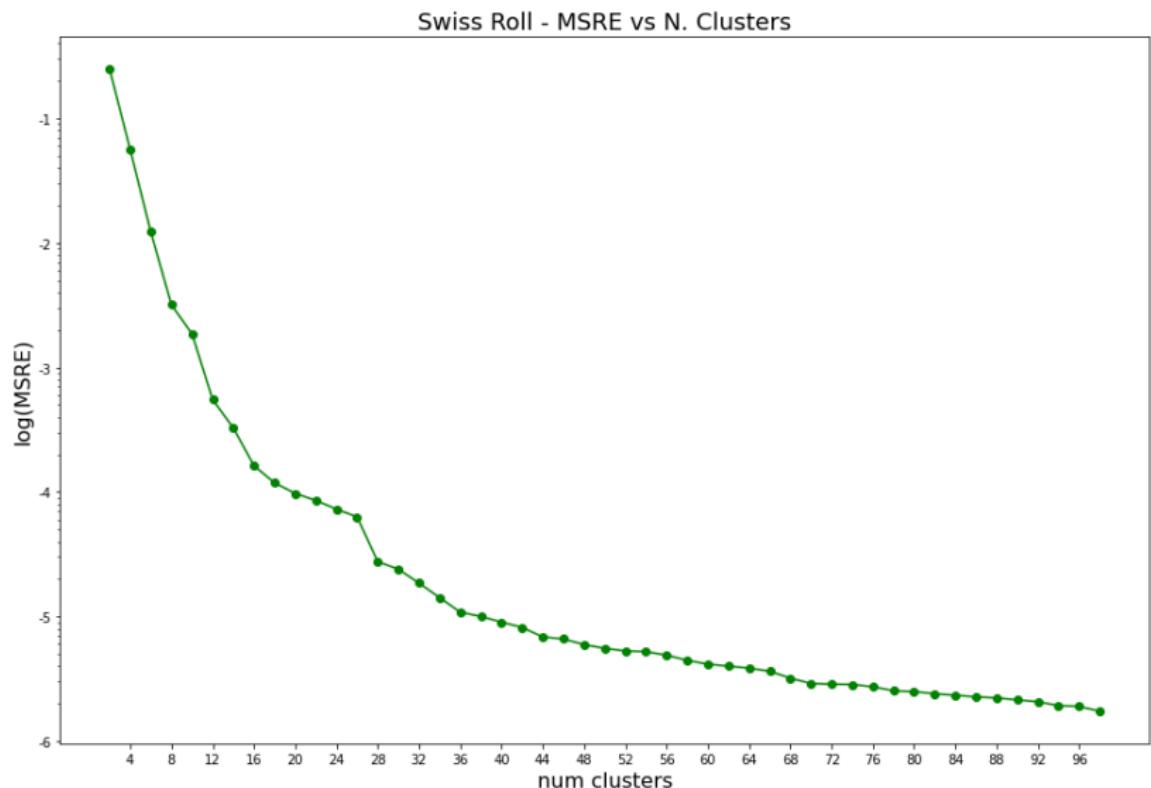
# Cluster finali S-Curve





# Cluster finali Swiss Roll

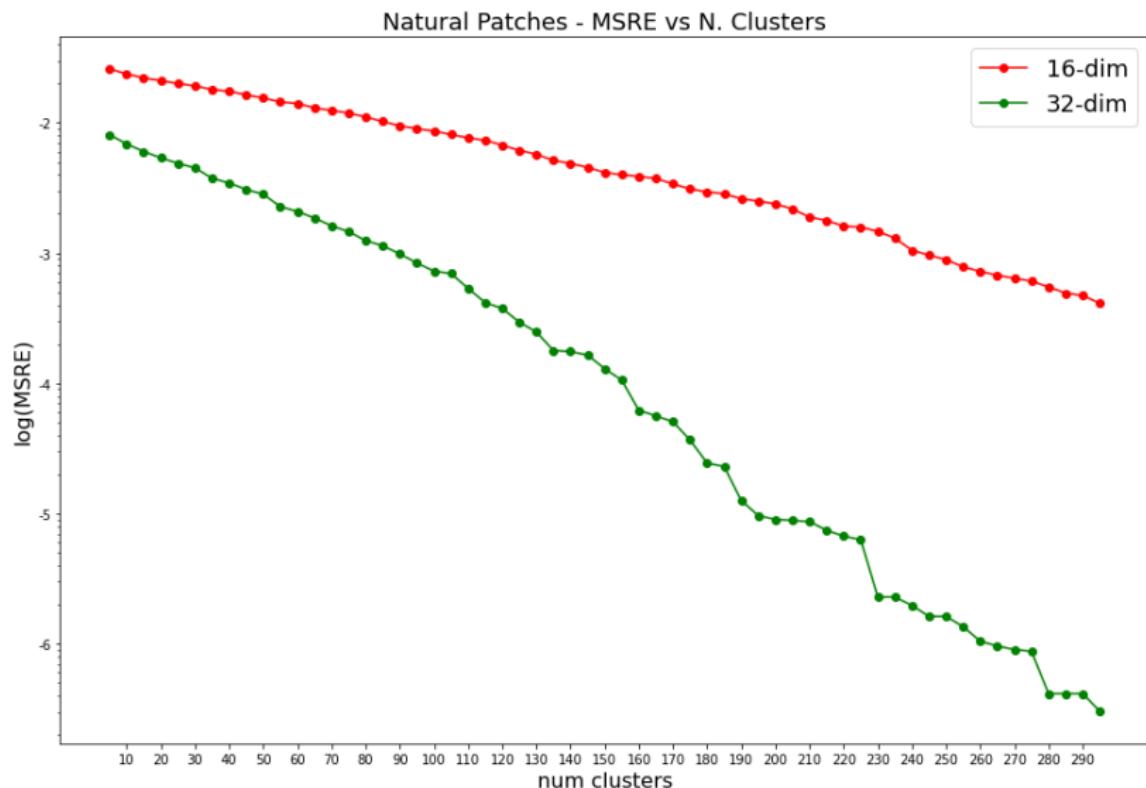




Un ulteriore test condotto è sulla ricostruzione di patch di immagini naturali che si assume appartengano ad una varietà di dimensione inferiore nel lavoro di

*Ramamurthy, et al. – Improved sparse coding using manifold projections.*

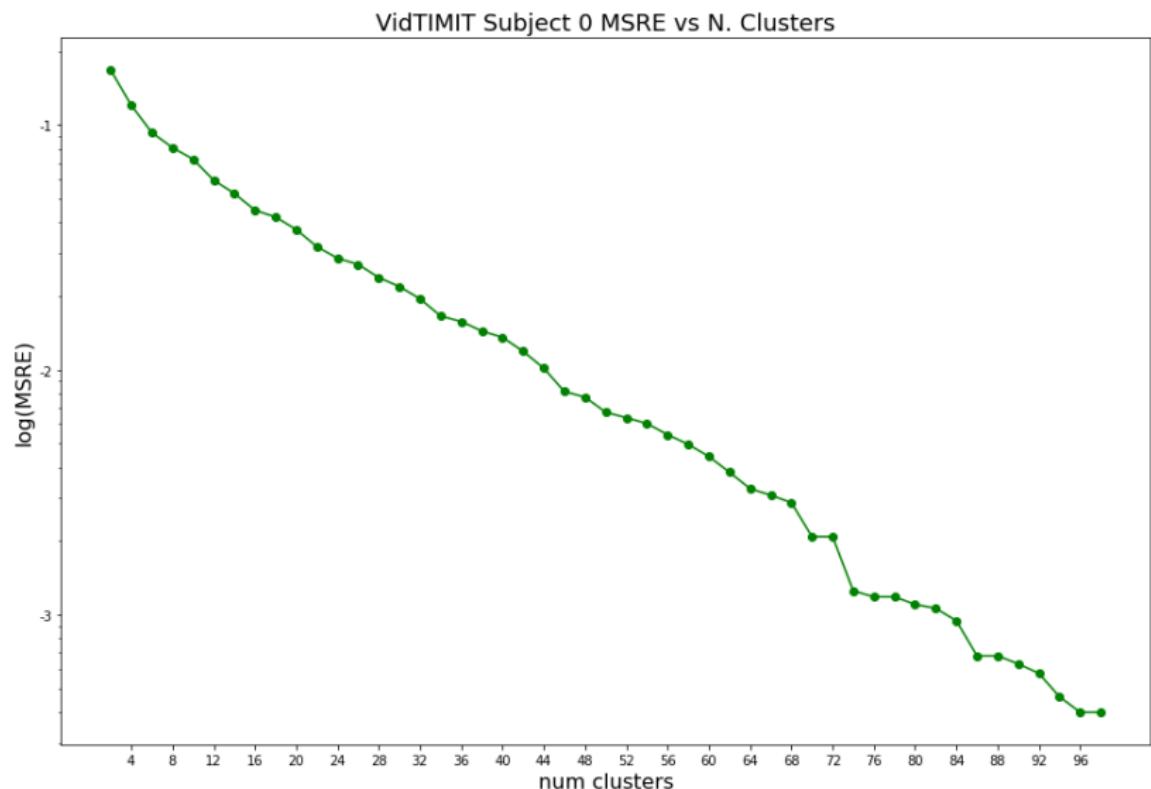
Il dataset utilizzato è il Berkeley Segmentation Dataset (BSDS), le patch sono di dimensione 8x8 e i test condotti con 5000 punti e k=30.



Sempre su immagini si è valutato il potere ricostruttivo dell'algoritmo su un dataset di facce prese dal VidTIMIT database. Le facce di un singolo soggetto che sposta la testa in varie direzioni sono state estratte utilizzando un algoritmo di face detection classico sviluppato da *P. Viola* e successivamente ridimensionate a 26x26 pixel.

Assumendo che le faccia in diverse pose di una stessa persona appartenga ad una varietà di dimensione inferiore si è applicato l'ACDT per approssimare tale varietà; il numero di punti di training variano da soggetto a soggetto, mediamente sono 750,  $k=15$  e la dimensione  $d=10$  (dove  $\mathbb{R}^N = \mathbb{R}^{676}$ ).

Le ricostruzioni sono fatte per  $\mathcal{L} = 12$ .



# Ricostruzione



# Cluster Risultanti



# Ricostruzione



# Cluster Risultanti



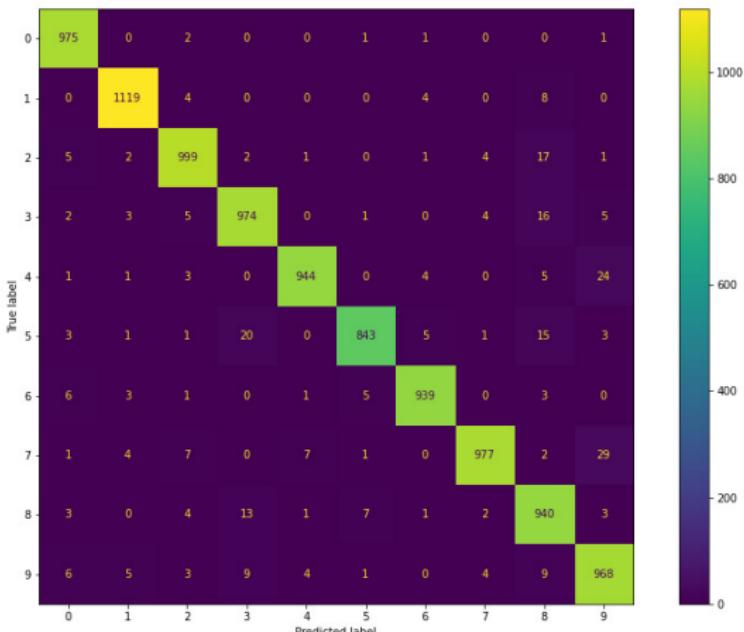
L'ultimo test condotto è sul MNIST, dataset di digit scritte a mano di dimensione 28x28. Per valutare la potenza dell'algoritmo al fine della classificazione si è valutato un indice chiamato *Geometric Separability Index* presentato nel paper di *C.Thornton, Separability is a learner's best friend.* Questo indice è definito come il rateo di coppie di neighbors che condividono la stessa classe. Il test è condotto con 2000 punti casuali per ogni digit  $k=15$  e  $d=100$ . Il GSI viene valutato nello spazio originale e nello spazio delle distanze dagli spazi affini trovati.

| Original 784 space | ACDT 100-d space $\mathcal{L} = 10$ |
|--------------------|-------------------------------------|
| 0.96               | 0.94                                |

# Precisione e Matrice di Confusione

Riportiamo anche l'accuracy e la confusion matrix relativa alla classificazione di 10000 punti del validation set del MNIST per  $\mathcal{L} = 12$ .

Accuracy: 96.78%



## Proposizione

$$\sum_{x \in C_i} D_T^2(M_x, M_{C_i \cup C_j}) \leq \sum_{x \in C_i} D_T^2(M_x, M_{C_j})$$

## Dimostrazione.

Supponiamo per assurdo che

$$\begin{aligned} \sum_{x \in C_i} D_T^2(M_x, M_{C_i \cup C_j}) &\geq \sum_{x \in C_i} D_T^2(M_x, M_{C_j}) \\ \implies \sum_{x \in C_i \cup C_j} D_T^2(M_x, M_{C_i \cup C_j}) &\geq \sum_{x \in C_i \cup C_j} D_T^2(M_x, M_{C_j}) \end{aligned}$$

Ma questo contraddice la definizione di  $M_{C_i \cup C_j}$ .

□

Passaggi per il calcolo di  $\tilde{d}$ , ricordiamo

$$\begin{aligned} d(C_i, C_j) &= p(C_i \cup C_j) - p(C_i) - p(C_j) \\ &= \sum_{x \in C_i} \left[ D_T^2 \left( M_x, M_{C_i \cup C_j} \right) - D_T^2 \left( M_x, M_{C_i} \right) \right] \\ &\quad + \sum_{x \in C_j} \left[ D_T^2 \left( M_x, M_{C_i \cup C_j} \right) - D_T^2 \left( M_x, M_{C_j} \right) \right] \end{aligned}$$

Sappiamo  $\sum_{x \in C_i} D_T^2 \left( M_x, M_{C_i \cup C_j} \right) \leq \sum_{x \in C_i} D_T^2 \left( M_x, M_{C_j} \right)$

Lo stesso vale naturalmente per  $C_j$ , sostituendo in  $d$  otteniamo:

$$\begin{aligned} d(C_i, C_j) &\leq \sum_{x \in C_i} \left[ D_T^2 \left( M_x, M_{C_j} \right) - D_T^2 \left( M_x, M_{C_i} \right) \right] \\ &\quad + \sum_{x \in C_j} \left[ D_T^2 \left( M_x, M_{C_i} \right) - D_T^2 \left( M_x, M_{C_j} \right) \right] \end{aligned}$$

## Appendice (2)

Ora per la diseguaglianza triangolare si ha

$$\begin{aligned} D_T(M_x, M_{C_j}) &\leq D_T(M_x, M_{C_i}) + D_T(M_{C_i}, M_{C_j}), \quad \forall x \in X \\ D_T(M_x, M_{C_i}) &\leq D_T(M_x, M_{C_j}) + D_T(M_{C_i}, M_{C_j}), \quad \forall x \in X \end{aligned}$$

E quindi elevando al quadrato e sommando sui punti di  $C_i$  e  $C_j$  rispettivamente otteniamo

$$\begin{aligned} &\sum_{x \in C_i} \left[ D_T^2(M_x, M_{C_j}) - D_T^2(M_x, M_{C_i}) \right] \\ &\leq 2D_T(M_{C_i}, M_{C_j}) \sum_{x \in C_i} D_T(M_x, M_{C_i}) + |C_i| D_T^2(M_{C_i}, M_{C_j}) \\ &\sum_{x \in C_j} \left[ D_T^2(M_x, M_{C_i}) - D_T^2(M_x, M_{C_j}) \right] \\ &\leq 2D_T(M_{C_i}, M_{C_j}) \sum_{x \in C_j} D_T(M_x, M_{C_j}) + |C_j| D_T^2(M_{C_i}, M_{C_j}) \end{aligned}$$

Otteniamo così la formula finale per la nostra dissimilarità approssimata  $\tilde{d}$  che dipende solo da informazioni pregresse

$$\begin{aligned} d(C_i, C_j) &\leq (|C_i| + |C_j|) D_T^2(M_{C_i}, M_{C_j}) \\ &+ 2D_T(M_{C_i}, M_{C_j}) \left[ \sum_{x \in C_i} D_T(M_x, M_{C_i}) + \sum_{x \in C_j} D_T(M_x, M_{C_j}) \right] \end{aligned}$$

### Proposizione

$$d \geq 0$$

### Dimostrazione.

$$\begin{aligned} d(C_i, C_j) &= \sum_{x \in C_i \cup C_j} D_T^2(M_x, M_{C_i \cup C_j}) - \sum_{x \in C_i} D_T^2(M_x, M_{C_i}) \\ &\quad - \sum_{x \in C_j} D_T^2(M_x, M_{C_j}) \\ &= \sum_{x \in C_i} \left[ D_T^2(M_x, M_{C_i \cup C_j}) - D_T^2(M_x, M_{C_i}) \right] \\ &\quad + \sum_{x \in C_j} \left[ D_T^2(M_x, M_{C_i \cup C_j}) - D_T^2(M_x, M_{C_j}) \right] \end{aligned}$$

Da cui segue la tesi data la definizione di  $M_{C_i}$ ,  $M_{C_j}$  e  $M_{C_i \cup C_j}$ .

□