Tangent-based manifold approximation with locally linear models

Sofia Karygianni, Pascal Frossard

Ecole Polytechnique Fédérale de Lausanne (EPFL), Signal Processing Laboratory (LTS4), CH-1015 Lausanne, Switzerland

introduzione

Motivati dal fatto che dati di dimensione alta siano comunemente difficili da trattare (The curse of dimensionality) si osserva che spesso segnali complessi possiedono una struttura sottostante che puo' permettere rappresentazioni adeguate in dimensione inferiore.

Un esempio e' dato da immagini catturate da diversi punti di vista di una stessa scena tridimensionale.

Nel dettaglio

In generale le varieta' (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 varieta' d-dimensionali (differenziabili) immerse in uno spazio Euclideo di dimensione $\mathbb{R}^N$ con $N \gg d$. Intuitivamente si puo' pensare a una varieta' 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 $\mathbb{R}^2$ che sono rispettivamente due varieta' di dimensione 2 e 1.

In questo lavoro si cerchera' di approssimare una varieta' arbitraria attraverso un modello semplice e computazionalmente efficiente ossia un set di sottospazi affini di dimensione fissata cercando di preservare globalmente la geometria della varieta' stessa.

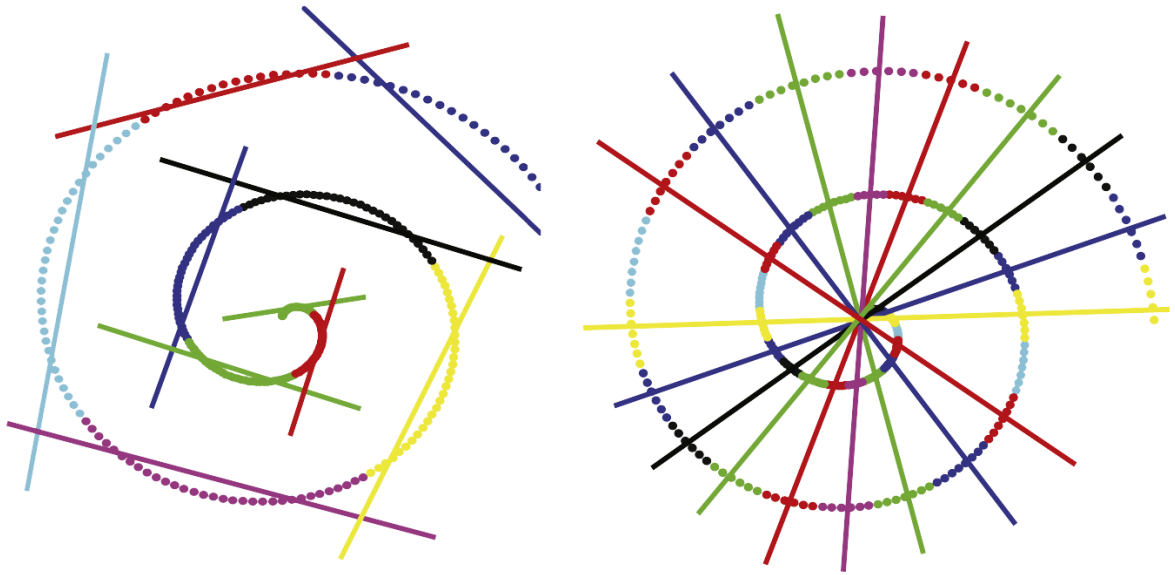


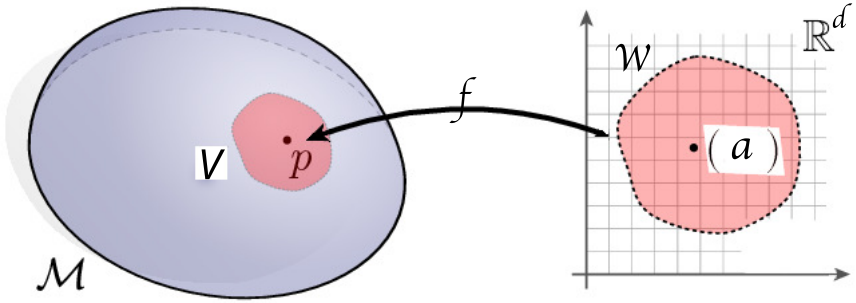
Figura 1: A sinistra una buona approssimazione di una varieta' 1D immersa in $\mathbb{R}^2$ dove i colori rappresentano i diversi gruppi di punti approssimati da linee. A destra una cattiva approssimazione che non preserva la geometria della varieta'.

Framework

Def. 1 (Varieta'). Un insieme $M \subseteq \mathbb{R}^N$ e' detto varieta' 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 $f : W \to \mathbb{R}^N$ iniettiva, differenziabile e ad inversa continua t.c.

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

Supponiamo $f(a) = x$ allora la matrice $Df(a)$ e la corrispondente trasformazione lineare $f_* : \mathbb{R}^d_a \to \mathbb{R}^N_x$ definiscono un sottospazio di dimensione d detto lo spazio tangente a M in x denotato M_x.
Per comodita' indicheremo da qui in poi con M_x lo spazio tangente ad M in x traslato pero' all'origine di $\mathbb{R}^N$.



Osservazione motivante

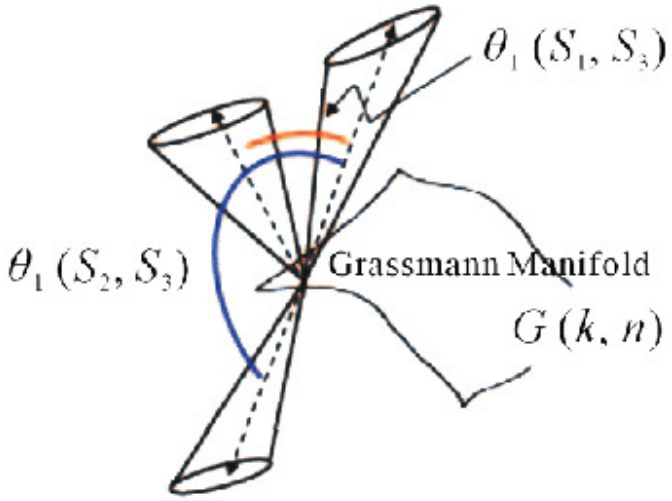
E' fondamentale osservare che se per opportuni x, V e W f e' lineare allora $Df(a) = Df(b)\ \forall a, b \in W$ e dunque lo spazio tangente di tutti i punti in $M \cap V$ coincide (a meno di una traslazione) e la regione di spazio puo' essere perfettamente rappresentata con dei sottospazi affini. Siamo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

Grassmann Manifold e Geodetiche

L'insieme dei sottospazi lineari di dimensione d di $\mathbb{R}^N$ e' detto Grassmanniana indicato $G_{N,d}(\mathbb{R}^N)$. In $G_{N,d}(\mathbb{R}^N)$ e' 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

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

dove $\theta = \{\theta_1, \ldots, \theta_d\}$ e' il vettore degli angoli principali di M_x e M_y.



Punto medio di Karcher

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

Def. 2 (Punto medio di Karcher su $G_{N,d}(\mathbb{R}^N)$ con distanza geodetica)**.** Il punto medio di un insieme C di punti di $G_{N,d}(\mathbb{R}^N)$ rispetto alla distanza D_T e' dato da

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

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).

Clustering

Sia $\mathcal{X} = \{x_k \in \mathbb{R}^N, k \in [1, m]\}$ la rappresentazione di una varieta' attraverso un insieme di suoi punti. Sia $G_{\mathcal{X}} = G(\mathcal{X}, E)$ il grafo simmetrico non orientato che rappresenta la geometria della varieta'. 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 varieta'.

Def. 3 (Partizione)**.** $\mathbf{C}_{\mathcal{L}}$ e' una partizione se $C_i \cap C_j = \emptyset \ \forall i \neq j, \ i, j \in \{1, ..., \mathcal{L}\}$ e $\bigcup_i C_i = \mathcal{X}$.

Non tutte le partizione saranno pero' valide una condizione sufficiente e' che la partizione sia formata solo da cluster con sottografi connessi.

Def. 4 (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 e' connesso se ogni coppia di nodi $x_i, x_j \in C_i$ e' connessa.

Validita'

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

Def. 5 (Predicato di *validita'*). $\Phi_{\mathcal{X}}(\mathbf{C}_{\mathcal{L}}) \equiv \mathbf{C}_{\mathcal{L}} \in \Phi_{\mathcal{L}}(\mathcal{X})$ definito come

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

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

Fondibilita'

Possiamo definire anche un predicato di fondibilita' Ψ che descrive se due cluster possono essere uniti dando vita a una partizione ancora valida, i predicati di fondibilita' e validita' sono legati dalla relazione seguente:
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.

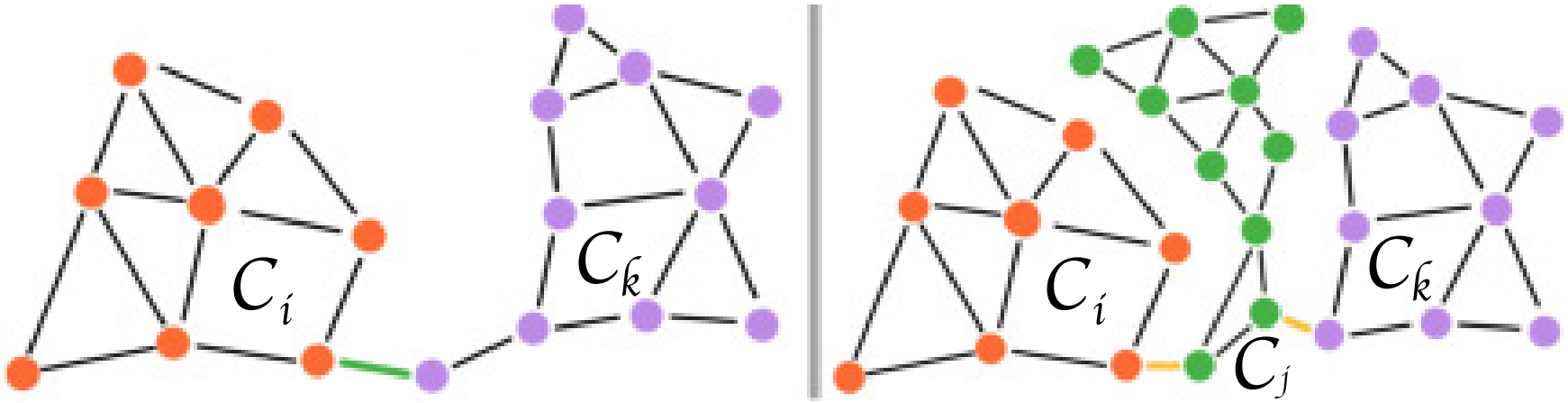


Figura 2: A sinistra due cluster C_i e C_k connessi e dunque fondibili. A destra tre cluster dove sia C_i che C_k sono fondibili con C_j ma C_i e C_k non sono fondibili.

E' facile osservare che fondendo le coppie (fondibili) in entrambe le configurazioni la partizione risultante e' composta da sottografi connessi e dunque valida.

Bonta' di una partizione

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

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

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

Osserviamo che per definizione $\mathcal{P}$ e' distributiva sui cluster e non negativa, inoltre e' nulla sui cluster formati da singoletti.

Il problema di approssimazione

Ora che abbiamo una misura di bonta' possiamo definire il nostro problema di approssimazione come la ricerca della partizione $\mathbf{C}^*_{\mathcal{L}}(\mathcal{X})$ dove

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

Verso una formulazione greedy

Vogliamo formulare una versione meno accurata ma piu' efficiente, introduciamo alcuni rilassamenti al problema andando a determinare un algoritmo greedy e bottom-up, ci serve pero' un modo per valutare la bonta' di una fusione di cluster.

Def. 6 (Misura di Dissimilarita'). $d : (C_i, C_j) \to R_0^+$
$d(C_i, C_j) = p(C_i \cup C_j) - p(C_i) - p(C_j)$

Esplicitando possiamo osservare che $d \geq 0$ infatti

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

Da cui segue la tesi data la definizione di M_{C_i}, M_{C_j} e $M_{C_i \cup C_j}$.
Possiamo allora riscrivere il nostro problema nella forma

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

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

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

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

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

Questo riduce notevolmente il costo computazionale ma non assicura piu' l'ottimalita' della soluzione trovata.

Siamo ora pronti a descrivere l'algoritmo di clustering.

Inizializzazione: $G_{\mathcal{X}}$ e M_x

$G_{\mathcal{X}}$ e' costruita connettendo ogni punto ai suoi vicini attraverso una *knn*, per ogni punto $x \in \mathcal{X}$ definiamo inoltre $N_x = \{y \in \mathcal{X} : (x, y) \in E\}$ l'insieme dei neighbors di x. Ora M_x puo' 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 $\left[N_x^0\right]$.

Algoritmo di merging

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 fondibilita' 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 dissimilarita' tra due cluster era definita come

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

Tale equazione e' costosa da computare poiche' richiede il calcolo di $M_{C_i \cup C_j}$ per ogni ipotetica fusione. Vorremmo una formulazione che dipenda solo da valori gia' 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 $\sum_{x \in C_i} D_T^2 \left(M_x, M_{C_i \cup C_j}\right) \leq \sum_{x \in C_i} D_T^2 \left(M_x, M_{C_j}\right)$, provabile p.a. Lo stesso vale naturalmente per C_j, allora sostituendo in d otteniamo:

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

Ora per la disuguaglianza triangolare si ha

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

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

$$\sum_{x \in C_i} \left[D_T^2 \left(M_x, M_{C_j} \right) - D_T^2 \left(M_x, M_{C_i} \right) \right]$$

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

$$\sum_{x \in C_j} \left[D_T^2 \left(M_x, M_{C_i} \right) - D_T^2 \left(M_x, M_{C_j} \right) \right]$$

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

Otteniamo cosi' la formula finale per la nostra dissimilarita' approssimata $\tilde{d}$ che dipende solo da informazioni pregresse

$$d \left(C_i, C_j \right) \leq \left(|C_i| + |C_j| \right) D_T^2 \left(M_{C_i}, M_{C_j} \right)$$

$$+ 2 D_T \left(M_{C_i}, M_{C_j} \right) \left[\sum_{x \in C_i} D_T \left(M_x, M_{C_i} \right) + \sum_{x \in C_j} D_T \left(M_x, M_{C_j} \right) \right]$$

Calcoleremo cosi' 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 $\left(C_i', C_j'\right) = \underset{\substack{C_i, C_j \in \mathbf{C}_{n-\lambda}^* \\ \psi(C_i, C_j) \text{ is true}}}{\operatorname{argmin}} \tilde{d}(C_i, C_j)$ ∗*Eq. (23)*∗

10: $\mathbf{C}_{n-\lambda+1}^* = (\mathbf{C}_{n-\lambda}^* \backslash \{C_i', C_j'\}) \cup \{C_i' \cup C_j'\}$

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

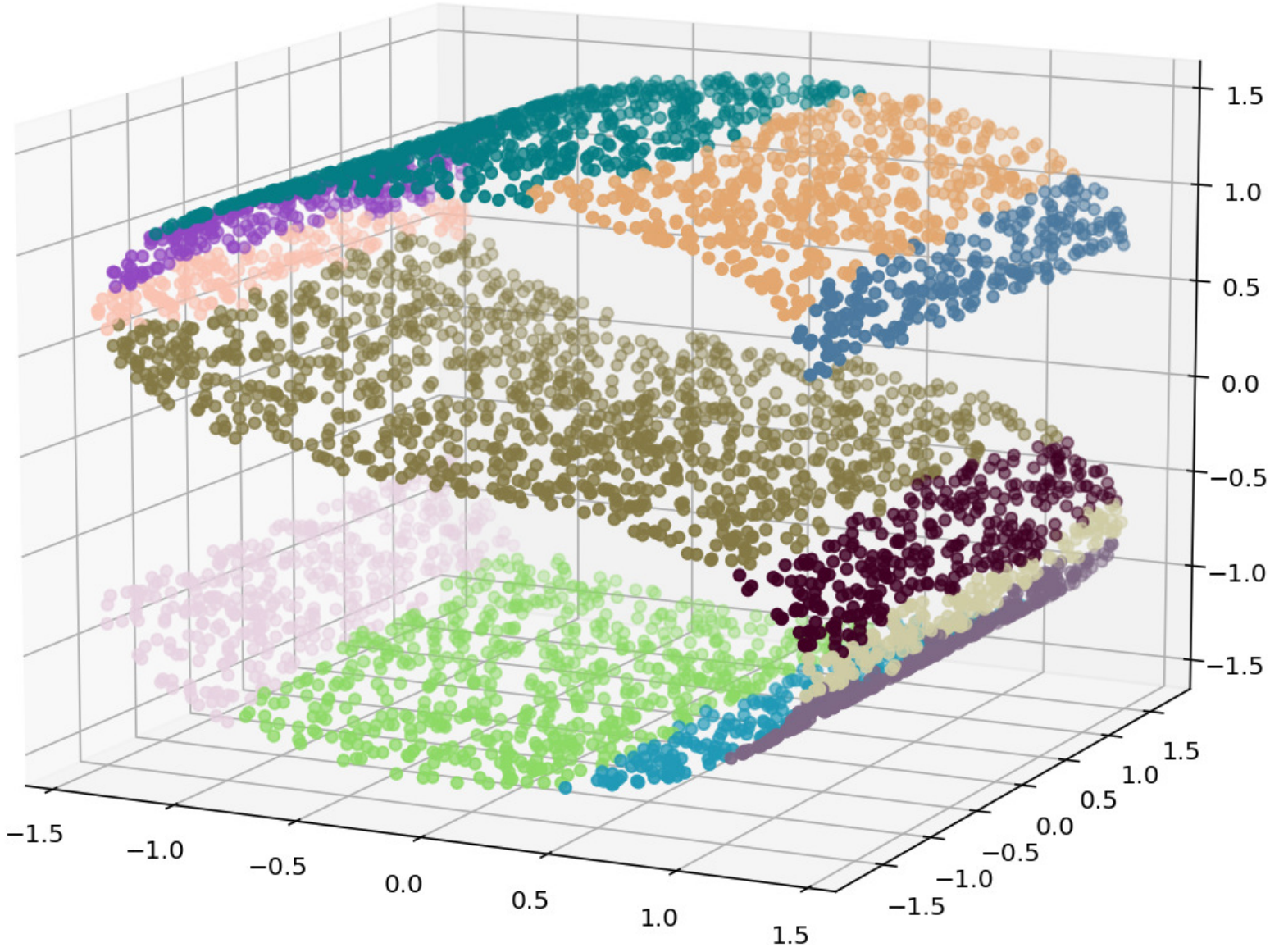
Nelle linee $14 - 16$ vengono computati i sottospazi affini finali attraverso SVD.

Risultati sperimentali

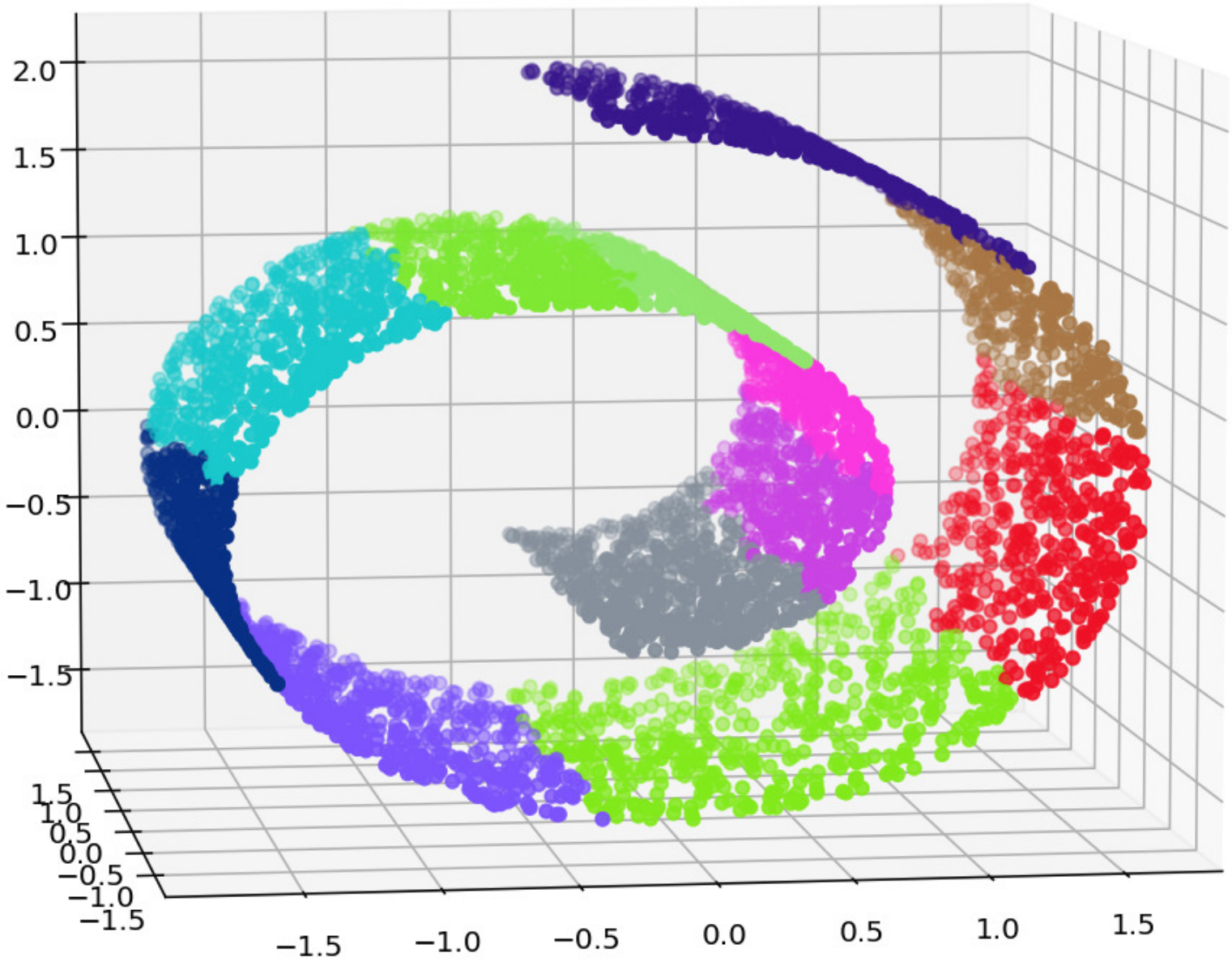
Iniziamo con l'approssimazione di 2 dataset sintetici 3D, la Swiss Roll e la S-Curve. I test sono condotti con 5000 punti, un neighborhood size di $k = 15$ e $d = 2$, la performance e' valutata con il *mean squared reconstruction error*. $MSRE = \frac{1}{N} \sum_{i=1}^{N} \|x_i - \hat{x}_i\|^2$.
Riportiamo anche i tempi di esecuzione dell'algoritmo su tali dataset:

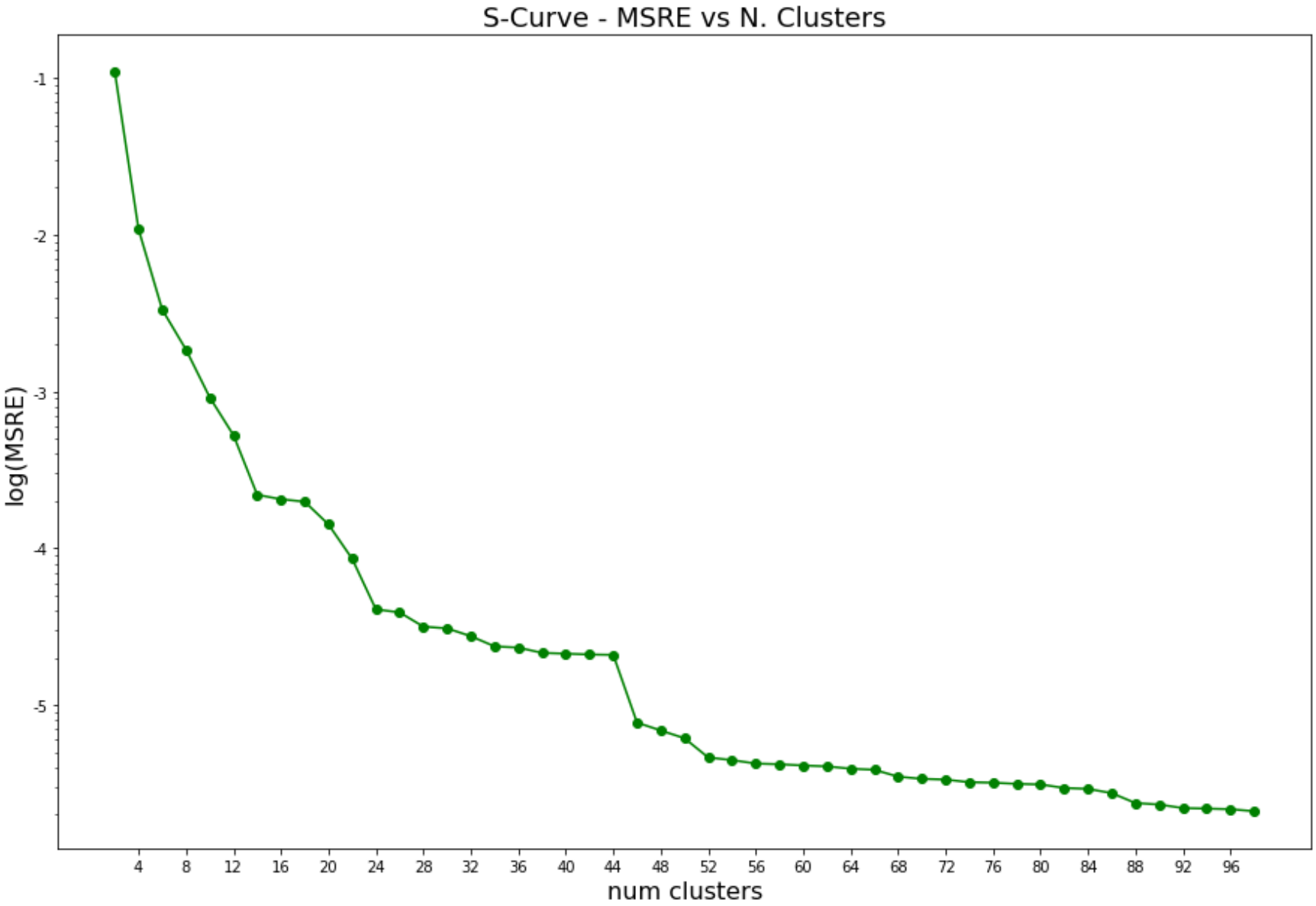
Paper - MBP 2.66Ghz	Mine - MBA 1.6Ghz	Mine - i5 3.60GHz
389s	267s	123s

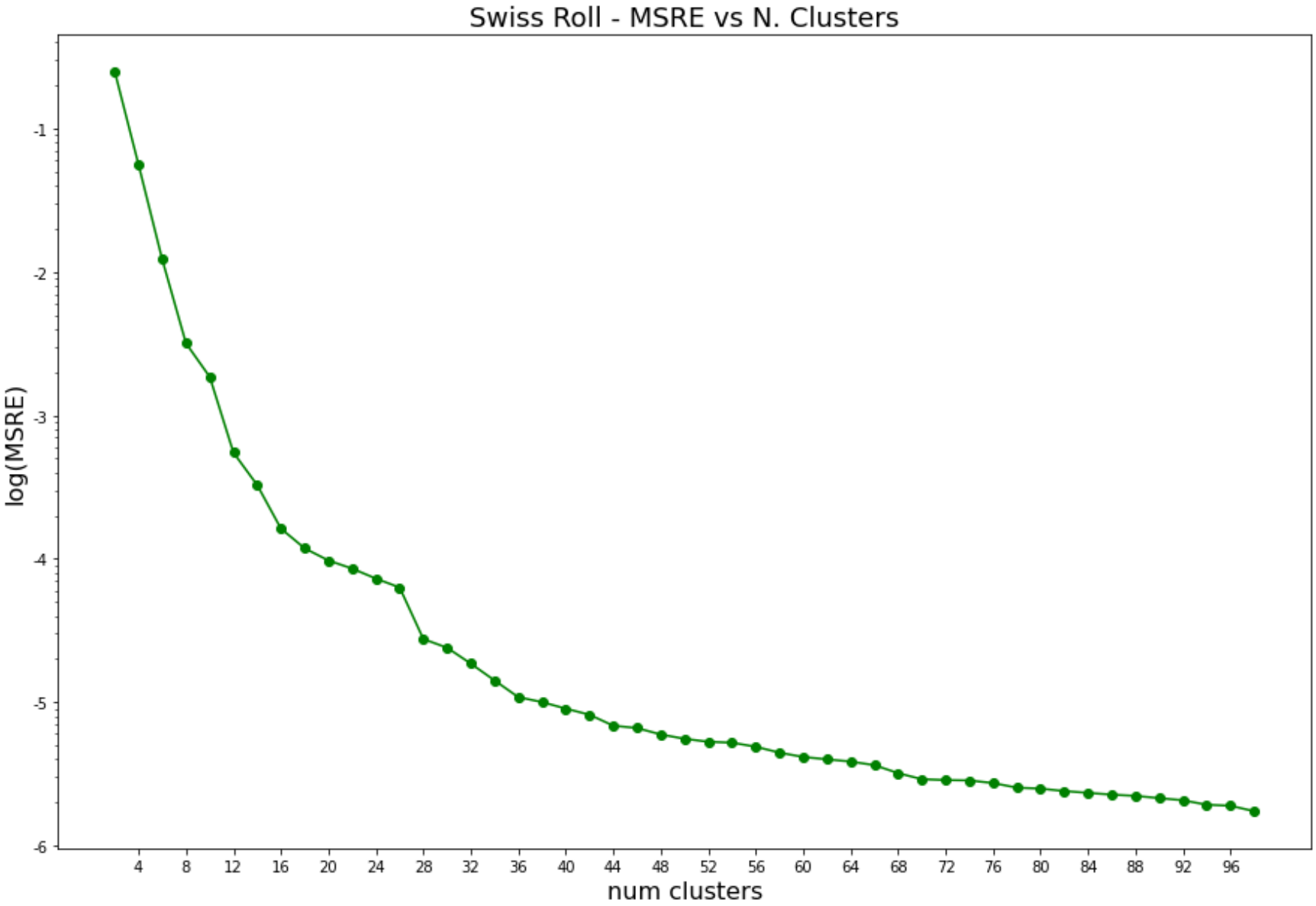
Cluster finali S-Curve



Cluster finali Swiss Roll




S-Curve - MSRE vs N. Clusters
log(MSRE)
-1
-2
-3
-4
-5
4
8
12
16
20
24
28
32
36
40
44
48
52
56
60
64
68
72
76
80
84
88
92
96
num clusters


Swiss Roll - MSRE vs N. Clusters
log(MSRE)
-1
-2
-3
-4
-5
-6
4
8
12
16
20
24
28
32
36
40
44
48
52
56
60
64
68
72
76
80
84
88
92
96
num clusters