PCA & t-SNE: A Visual Guide

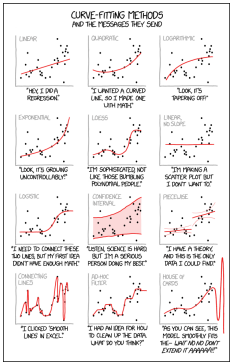
No Formulas, Just Pictures

Prof. Dr. Jörg Osterrieder

Methods and Algorithms — MSc Data Science

Spring 2026

When You Have Too Many Dimensions



Your dataset has 50 columns.

You can only plot 2 axes.

What do you do?

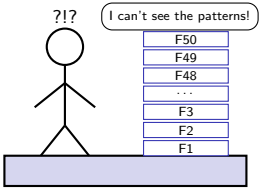
XKCD #2048 by Randall Munroe (CC BY-NC 2.5)

What You Will Learn Today

1. Explain what PCA does — without any formulas
2. Describe when t-SNE beats PCA
3. Choose the right method for a given dataset

No formulas in this lecture. Everything is explained with pictures and analogies.

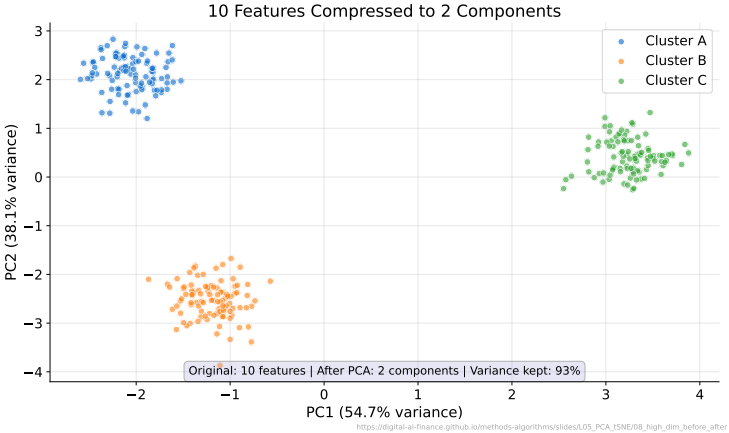
The Problem: Too Many Features!



Imagine a bank tracking 50 risk metrics per portfolio. How do you visualize this?

High-dimensional data is everywhere: genomics (20k genes), NLP (50k words), finance (hundreds of factors).

Compression Works!



- Each dot was originally described by 10 numbers (features)
- PCA compressed it to just 2 numbers
- The three clusters are still clearly visible

PCA found the 2 directions that preserve the most spread — no human picked them.

Why Not Just Pick Two Features?

Cherry-Picking

- Pick 2 out of 50 features
- Lose 48 features' information
- Which pair is best? Nobody knows

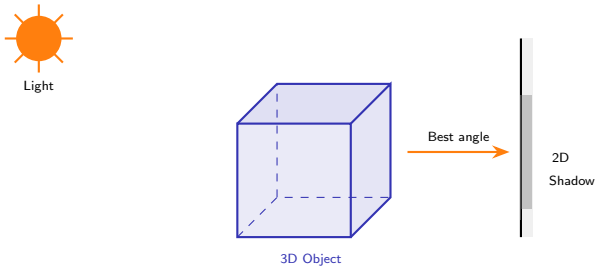
PCA Approach

- Blend ALL features into 2 new axes
- Automatic — no guessing
- Keeps maximum information

PCA does not discard features — it blends them.

Selecting features = you choose. PCA = the data chooses the best blend.

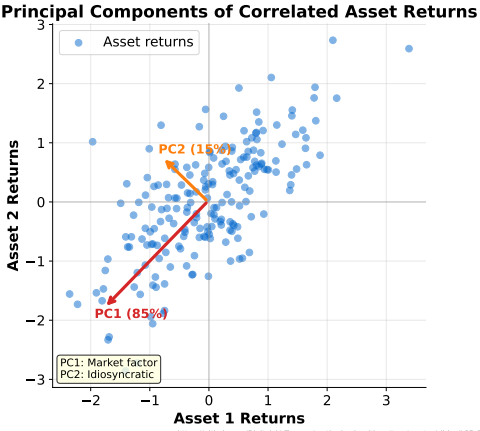
PCA = Finding the Best Shadow



PCA finds the angle that makes the shadow show the most detail.

PCA is literally a projection: it casts your high-dimensional data onto a lower-dimensional "wall."

PCA Finds the Best Directions

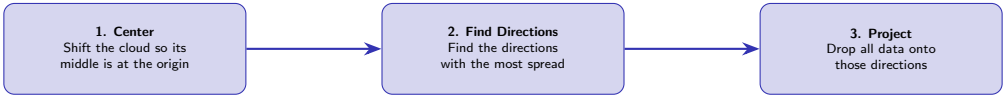


https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L05_PCA_tSNE/02_principal_components

- The first arrow (PC1) points along the main cloud — maximum spread
- The second arrow (PC2) is perpendicular — remaining spread
- Together they form new coordinate axes

PCA is a rotation: it views your data from the angle that shows the most structure.

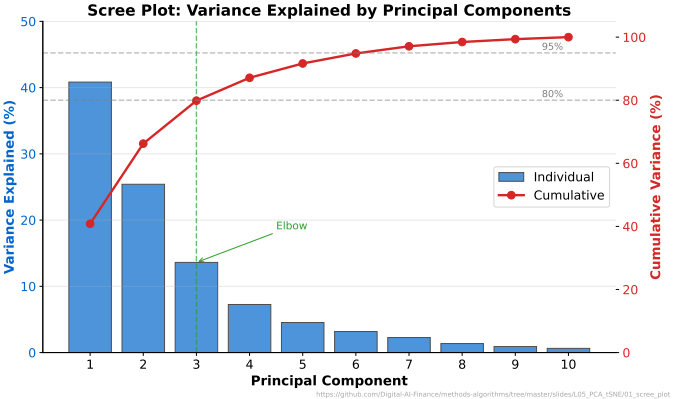
PCA in Three Steps



That's the entire algorithm. Three steps.

Step 2 uses eigenvalues internally, but you never need to compute them by hand — sklearn does it.

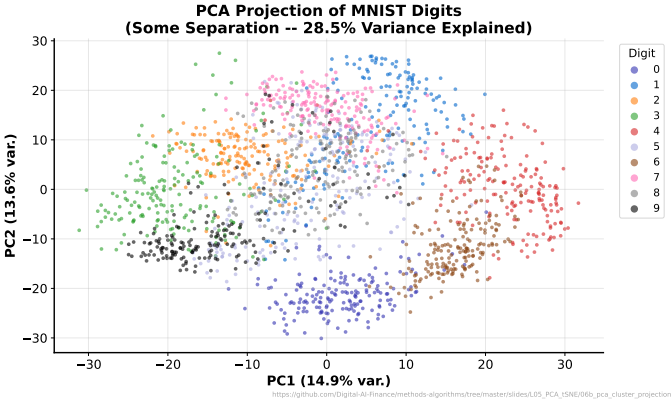
How Many Components Do We Keep?



- Bars = variance captured by each component (first few are tall)
- Line = cumulative total (crosses 80% quickly)
- Stop at the "elbow" — where adding more barely helps

Named after geological "scree" (cliff rubble). Stop where the cliff flattens into rubble.

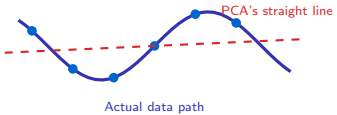
PCA Separates Some Clusters. . .



- Each color is a different group
- PCA separates some groups well
- But overlapping groups stay mixed — PCA is linear

PCA works great when clusters are separated by straight lines. But what if the data curves?

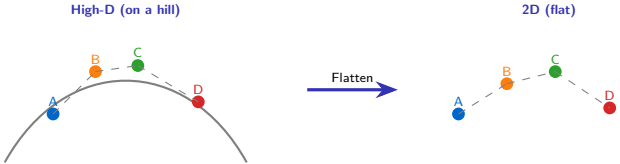
The Limit of Straight Lines



- PCA uses straight lines (linear) — great for flat data
- But real data often curves — we need something smarter...

When data lies on a curved surface, PCA cuts through it instead of following the curve.

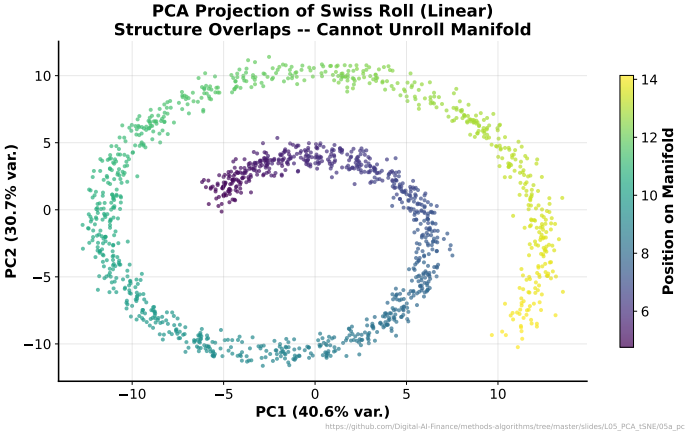
t-SNE: Keep Your Neighbors Close



t-SNE flattens the data while keeping nearby points nearby.

t-SNE = t-distributed Stochastic Neighbor Embedding. The key idea: preserve local neighborhoods.

PCA Fails on Curves: The Swiss Roll

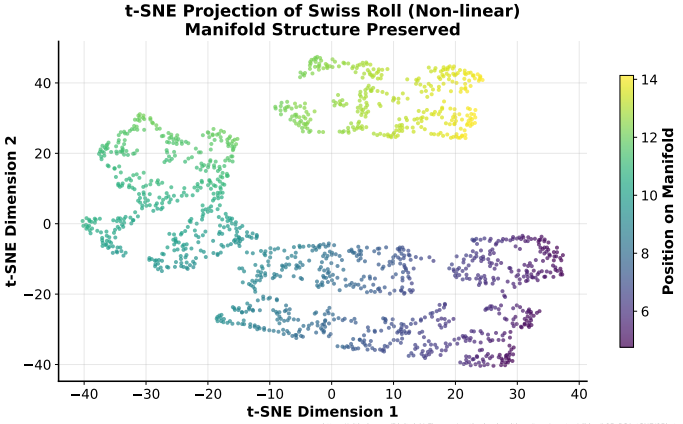


https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L05_PCA_tSNE/05a_pca_swiss_roll

- The data spirals in 3D like a rolled carpet
- PCA smashes it flat — distant points end up next to each other

PCA projects onto a plane, collapsing the spiral and mixing distant points.

t-SNE Unrolls the Carpet!



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L05_PCA_tSNE/05b_tsne_swiss_roll

- t-SNE preserves local distances along the spiral
- Points that were neighbors on the roll stay neighbors in 2D

t-SNE successfully "unrolls" the manifold by preserving local neighbourhood structure.

How t-SNE Works (No Math)

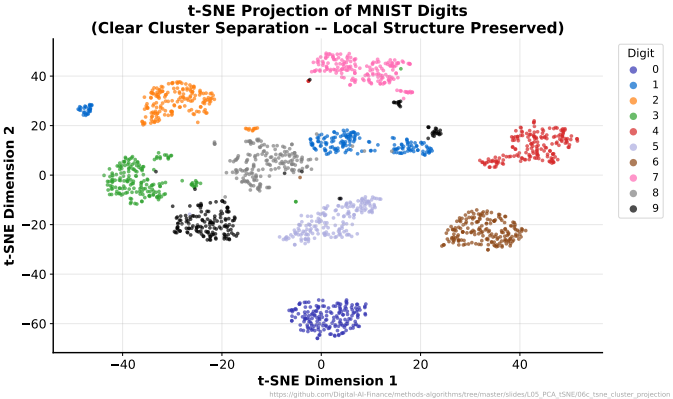
1. **Measure neighborhoods** — for each point, find its nearest neighbors
2. **Scatter randomly** — place all points on a 2D canvas at random
3. **Nudge until right** — move points until 2D neighborhoods match the originals

Intuition

Think of it as a spring system — connected points pull each other close.

t-SNE iterates hundreds of times, adjusting positions until the layout stabilizes.

t-SNE: Much Better Cluster Separation!



- Compare this with the PCA version (slide 11)
- t-SNE pulls apart clusters that PCA left overlapping
- The groups are now clearly separated

t-SNE excels at revealing clusters. But warning: inter-cluster distances are NOT meaningful.

Three Things to Watch Out For

1. ! **Distances between clusters mean nothing**
   Only within-cluster structure is reliable
2. ! **Run it multiple times**
   Different random seeds give different layouts
3. ! **Slow on big data**
   Use PCA first to reduce to 50 dimensions, then run t-SNE

t-SNE is for visualization only. Never use t-SNE coordinates as input features for a classifier.

PCA vs t-SNE: Quick Comparison

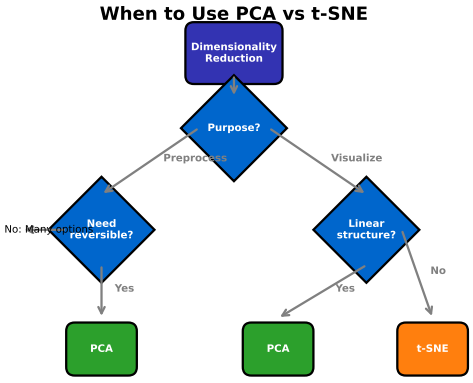
Aspect	PCA	t-SNE
Type	Linear	Non-linear
Preserves	Global structure	Local neighborhoods
Speed	Fast (seconds)	Slow (minutes)
Output	New features for ML	2D plot only
Deterministic	Yes	No (random seed)

They are complementary tools, not competitors.

Use PCA to preprocess, t-SNE to visualize.

In practice, run PCA to 50 dims first, then apply t-SNE for visualization.

Which Method Should I Use?



PCA: Fast, linear, reversible, for preprocessing

t-SNE: Slow, non-linear, visualization only, preserves local structure

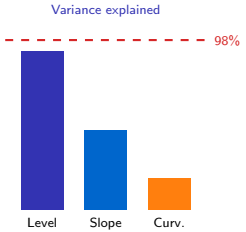
https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L05_PCA_tSNE/07_decision_flowchart

- Start at the top and follow the arrows
- In practice, try both and compare

When in doubt: run PCA first (fast, interpretable), then t-SNE for visualization.

Why Bankers Love PCA

- Bond markets track dozens of interest rates
- PCA reveals just 3 hidden factors:
  Level, Slope, Curvature
- These 3 factors explain over 98% of all bond price movements



Litterman & Scheinkman (1991): 3 PCA factors explain >98% of US Treasury yield curve movements.

Try It Yourself: 5 Lines of Python

PCA

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_reduced = pca.fit_transform(X)
```

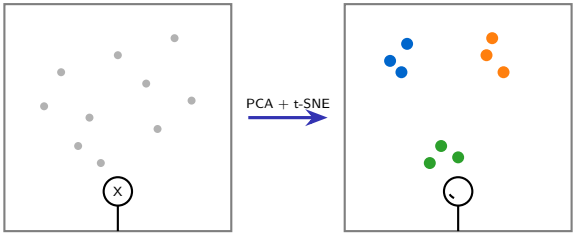
t-SNE

```
from sklearn.manifold import TSNE
tsne = TSNE(n_components=2)
X_embedded = tsne.fit_transform(X)
```

That's it. Three lines each. sklearn does the heavy lifting.

Always scale your data first: `StandardScaler().fit_transform(X)` **before PCA.**

From Chaos to Clarity



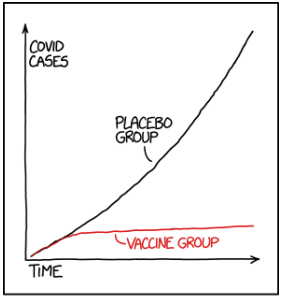
Dimensionality reduction turns overwhelming data into actionable visualizations.

Five Things to Remember

1. High-dimensional data is hard to see — PCA and t-SNE make it visible
2. PCA blends features into new axes that capture maximum spread
3. t-SNE preserves neighborhoods — great for finding clusters
4. PCA is fast and gives you new features; t-SNE is slow but reveals hidden groups
5. In finance, 3 PCA factors explain nearly all bond market movement

PCA for preprocessing and speed. t-SNE for visualization and cluster discovery. Use both.

Go Find the Signal in the Noise



XKCD #2400 by Randall Munroe (CC BY-NC 2.5). Next: try the Jupyter notebook with real financial data!