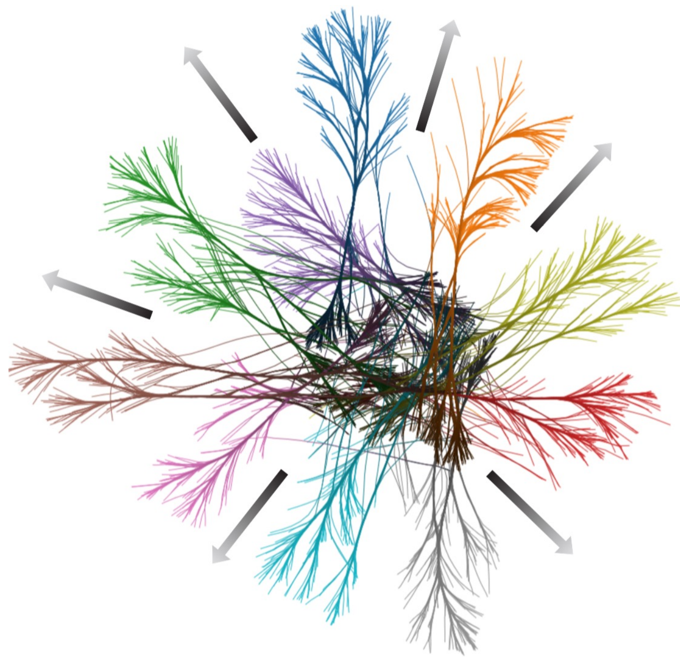
What is a Good Projection

And Why We Should Care About It

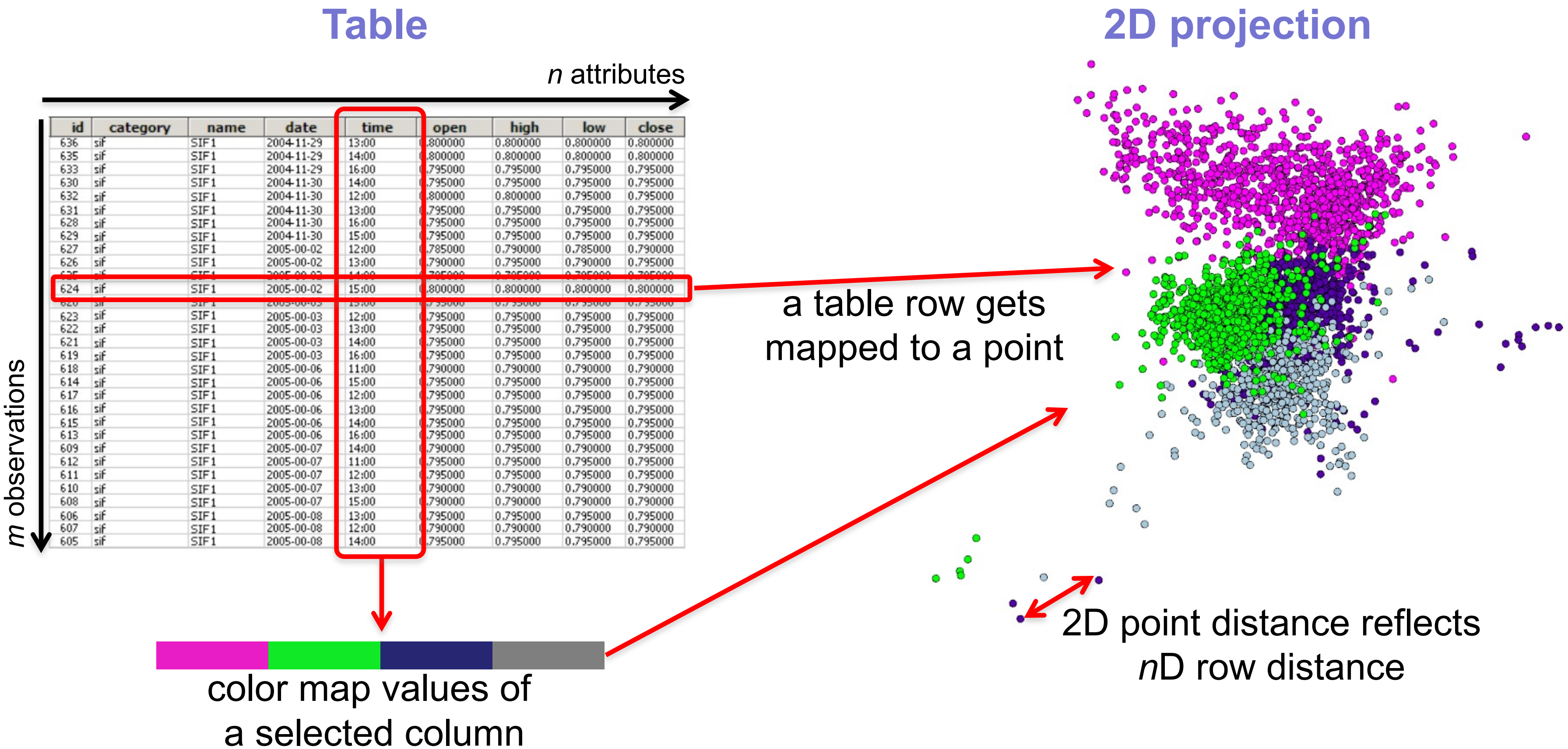


prof. dr. Alexandru (Alex) Telea

Department of Information and Computing Science
Utrecht University, the Netherlands

Utrecht University

Projections



id	category	name	date	time	open	high	low	close
636	sif	SIF1	2004-11-29	13:00	.800000	0.800000	0.800000	0.800000
635	sif	SIF1	2004-11-29	14:00	.800000	0.800000	0.800000	0.800000
633	sif	SIF1	2004-11-29	16:00	.795000	0.795000	0.795000	0.795000
630	sif	SIF1	2004-11-30	14:00	.795000	0.795000	0.795000	0.795000
632	sif	SIF1	2004-11-30	12:00	.800000	0.800000	0.795000	0.795000
631	sif	SIF1	2004-11-30	13:00	.795000	0.795000	0.795000	0.795000
628	sif	SIF1	2004-11-30	16:00	.795000	0.795000	0.795000	0.795000
629	sif	SIF1	2004-11-30	15:00	.795000	0.795000	0.795000	0.795000
627	sif	SIF1	2005-00-02	12:00	.785000	0.790000	0.785000	0.790000
626	sif	SIF1	2005-00-02	13:00	.790000	0.795000	0.790000	0.795000
624	sif	SIF1	2005-00-02	15:00	.800000	0.800000	0.800000	0.800000
623	sif	SIF1	2005-00-03	12:00	.795000	0.795000	0.795000	0.795000
622	sif	SIF1	2005-00-03	13:00	.795000	0.795000	0.795000	0.795000
621	sif	SIF1	2005-00-03	14:00	.795000	0.795000	0.795000	0.795000
619	sif	SIF1	2005-00-03	16:00	.795000	0.795000	0.795000	0.795000
618	sif	SIF1	2005-00-06	11:00	.790000	0.790000	0.790000	0.790000
614	sif	SIF1	2005-00-06	15:00	.795000	0.795000	0.795000	0.795000
617	sif	SIF1	2005-00-06	12:00	.795000	0.795000	0.795000	0.795000
616	sif	SIF1	2005-00-06	13:00	.795000	0.795000	0.795000	0.795000
615	sif	SIF1	2005-00-06	14:00	.795000	0.795000	0.795000	0.795000
613	sif	SIF1	2005-00-06	16:00	.795000	0.795000	0.795000	0.795000
609	sif	SIF1	2005-00-07	14:00	.790000	0.795000	0.790000	0.795000
612	sif	SIF1	2005-00-07	11:00	.795000	0.795000	0.795000	0.795000
611	sif	SIF1	2005-00-07	12:00	.795000	0.795000	0.795000	0.795000
610	sif	SIF1	2005-00-07	13:00	.790000	0.790000	0.790000	0.790000
608	sif	SIF1	2005-00-07	15:00	.790000	0.790000	0.790000	0.790000
606	sif	SIF1	2005-00-08	13:00	.795000	0.795000	0.795000	0.795000
607	sif	SIF1	2005-00-08	12:00	.790000	0.790000	0.790000	0.790000
605	sif	SIF1	2005-00-08	14:00	.795000	0.795000	0.795000	0.795000

Why is this useful?

- no matter how large *n* is, we obtain a 2D scatterplot-like image (so it's visually scalable)
- point-to-point distance (in 2D) shows similarity of observations (in *n*D)
- coloring points by one attribute can show additional information on the observations

Which projection technique is the "best"?

Projections are clearly useful tools for ML engineering
But which projection is the *best*?

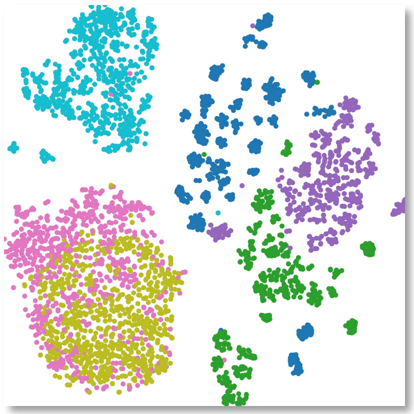
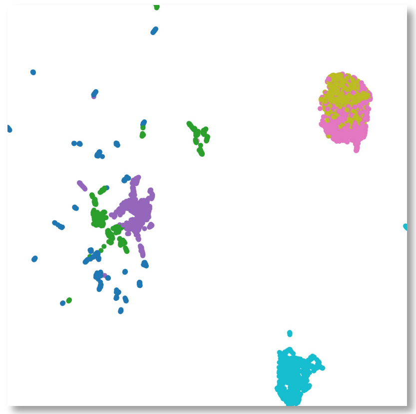
PCA: Poor separation...

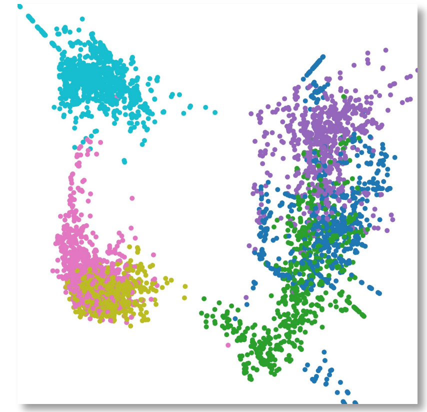
Isomap: Bit better separation...

t-SNE: Strong separation

UMAP: Extreme separation

Which projection technique is the "best"?



We can rephrase this question as:

Which projection is **correct**?

Which projection technique is the "best"?

Which projection is ~~correct~~?

An absolute judgement of correctness may not be possible (or even desirable)

Which projection technique is the "best"?

Rephrase: Which projection is more **faithful to the data**?

Enter Projection Quality Metrics (PQMs)!

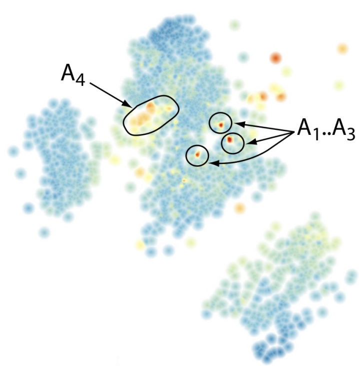
Projection Quality Metrics

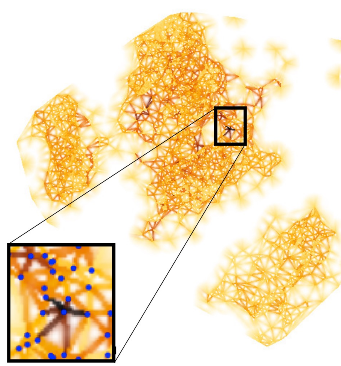
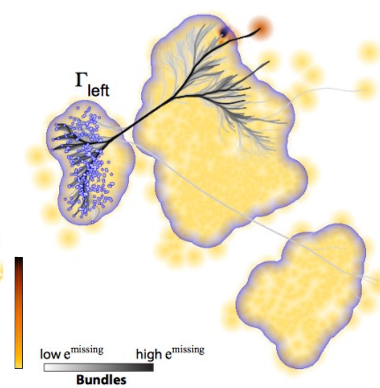
Functions $\mathcal{M}(\boldsymbol{X}, \boldsymbol{Y}, C)$ taking a dataset $\boldsymbol{X}$, its projection $\boldsymbol{Y}$, and possibly class info C

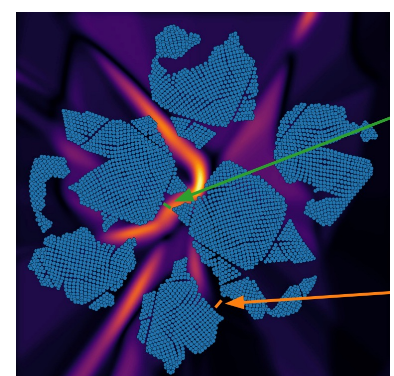
- Are pairwise distances **distorted** in $\boldsymbol{Y}$? → Stress
- Are pairwise distances in $\boldsymbol{Y}$ **correlated** to those in $\boldsymbol{X}$? → Shepard goodness
- Are **neighborhoods** in $\boldsymbol{Y}$ different than those in $\boldsymbol{X}$? → Trustworthiness/Continuity
- …

Dozens such metrics exist!

Metric	Definition	Type	Range
Trustworthiness (M_t)	$1 - \frac{2}{NK(2n-3K-1)} \sum_{i=1}^{N} \sum_{j \in U_i^{(K)}} (r(i,j) - K)$	scalar	$[0, \mathbf{1}]$
Continuity (M_c)	$1 - \frac{2}{NK(2n-3K-1)} \sum_{i=1}^{N} \sum_{j \in V_i^{(K)}} (\hat{r}(i,j) - K)$	scalar	$[0, \mathbf{1}]$
Normalized stress (M_σ)	$\frac{\sum_{ij}(\Delta^n(\mathbf{x}_i,\mathbf{x}_j) - \Delta^q(P(\mathbf{x}_i),P(\mathbf{x}_j)))^2}{\sum_{ij}\Delta^n(\mathbf{x}_i,\mathbf{x}_j)^2}$	scalar	$[\mathbf{0}, 1]$
Neighborhood hit (M_{NH})	$\sum_{i=1}^{N} \frac{\lvert j \in N_i^{(K)} : l_j = l_i \rvert}{KN}$	scalar	$[0, \mathbf{1}]$
Shepard diagram (S)	Scatterplot $(\lVert\mathbf{x}_i - \mathbf{x}_j\rVert, \lVert P(\mathbf{x}_i) - P(\mathbf{x}_j)\rVert), 1 \le i \le N, i \ne j$	point-pair	-
Shepard goodness (M_S)	Spearman rank correlation of Shepard diagram	scalar	$[0, \mathbf{1}]$
Average local error ($M_a(i)$)	$\frac{1}{N-1} \sum_{j \ne i} \left\lvert \frac{\Delta^n(\mathbf{x}_i,\mathbf{x}_j)}{\max_{i,j} \Delta^n(\mathbf{x}_i,\mathbf{x}_j)} - \frac{\Delta^q(P(\mathbf{x}_i),P(\mathbf{x}_j))}{\max_{i,j} \Delta^q(P(\mathbf{x}_i),P(\mathbf{x}_j))} \right\rvert$	local (per-point)	$[\mathbf{0}, 1]$





Which projections are *faithful to data*?

surveys

techniques

Projection Acronym	Projection Full Name	Fodor *et al.* [18]	Hoffman *et al.* [1]	Yin *et al.* [19]	Maaten *et al.* [13]	Bunte *et al.* [15]	Engel *et al.* [27]	Sorzano *et al.* [12]	Cunningham *et al.* [23]	Gisbrecht *et al.* [21]	Liu *et al.* [2]	Xie *et al.* [24]	Nonato *et al.* [10]	Ours
AE	Autoencoder				•									•
CCA	CCA (Canonical Correlations Analysis)								•					
CHL	Chalmers												•	
CLM	ClassiMap												•	
CuCA	CCA (Curvilinear Component Analysis)												•	
DM	Diffusion Maps				•									•
DML	Distance Metric Learning								•					
EM	Elastic Maps							•						
FA	Factor Analysis	•						•	•					•
FD	Force-Directed												•	
FMAP	FastMap												•	•
FS	Feature Selection											•		
GDA	Generalized Discriminant Analysis													•
GPLVM	Gaussian Process Latent Variable Model													•
GTM	Generative Topographic Mapping							•					•	
ICA	Independent Component Analysis	•						•	•					
F-ICA	FastICA													•
NL-ICA	Nonlinear ICA	•												
IDMAP	IDMAP													•
ISO	Isomap		•	•	•	•	•			•			•	•
L-ISO	Landmark Isomap													•
KECA	Kernel Entropy Component Analysis							•						
KLP	Kelp												•	
LAMP	LAMP												•	•
LDA	Linear Discriminant Analysis								•		•	•	•	
LE	Laplacian Eigenmaps				•	•				•	•		•	•
LLC	Locally Linear Coordination				•									•
LLE	Locally Linear Embedding		•	•	•	•	•			•	•		•	•
H-LLE	Hessian LLE				•									•
M-LLE	Modified LLE													•
LMNN	Large-Margin Nearest Neighbor Metric													•
LoCH	Local Convex Hull												•	
LPP	Locality Preserving Projection								•					•
LR	Linear Regression								•					
LSP	Least Square Projection												•	•
LTSA	Local Tangent Space Alignment				•								•	•
L-LTSA	Linear Local Tangent Space Alignment													•
MAF	Maximum Autocorrelation Factors								•					
MC	Manifold Charting				•					•				•
MCA	Multiple Correspondence Analysis												•	
MCML	Maximally Collapsing Metric Learning													•
MDS	Metric Multidimensional Scaling	•	•	•	•	•	•	•	•		•		•	•
L-MDS	Landmark MDS													•
MG-MDS	Multi-Grid MDS						•							
N-MDS	Nonmetric MDS (Kruskal)		•				•						•	•
ML	Manifold Learning						•							
MVU	Maximum Variance Unfolding				•	•				•			•	
FMVU	Fast MVU													
L-MVU	Landmark MVU													•
NeRV	Neighborhood Retrieval Visualizer					•								
t-NeRV	t-NeRV					•								
NMF	Nonnegative Matrix Factorization							•	•					•
NLM	Nonlinear Mapping												•	
NN	Neural Networks	•												
PBC	Projection By Clustering													•
PC	Principal Curves	•						•						
PCA	Principal Component Analysis	•	•		•		•	•	•	•	•	•	•	•
I-PCA	Incremental PCA							•						•
K-PCA-P	Kernel PCA (Polynomial)													•
K-PCA-R	Kernel PCA (RBF)		•		•		•	•		•				•
K-PCA-S	Kernel PCA (Sigmoid)													•
L-PCA	Localized PCA							•						
NL-PCA	Nonlinear PCA	•		•				•						
P-PCA	Probabilistic PCA								•					•
R-PCA	Robust PCA							•						
S-PCA	Sparse PCA							•						•
PLMP	Part-Linear Multidimensional Projection												•	
PLP	Piecewise Laplacian-based Projection						•						•	
PLSP	Piecewise Least Square Projection													•
PM	Principal Manifolds			•										
PP	Projection Pursuit	•												
RBF-MP	RBF Multidimensional Projection												•	
RP	Random Projections	•										•		
G-RP	Gaussian Random Projection													•
S-RP	Sparse Random Projection													•
SAM	Sammon Mapping				•									
R-SAM	Rapid Sammon (Pekalska)												•	•
SDR	Sufficient Dimensionality Reduction								•					
SFA	Slow Feature Analysis								•					
SMA	Smacof												•	
SNE	Stochastic Neighborhood Embedding					•							•	
T-SNE	t-Dist. Stochastic Neighborhood Embedding					•				•	•			•
SOM	Self-Organizing Maps	•		•				•					•	
ViSOM	ViSOM (Visualization-induced SOM)			•										
SPE	Stochastic Proximity Embedding													•
G-SVD	Generalized SVD							•						
T-SVD	Truncated SVD													•
TF	Tensor Factorization							•						
UMAP	Uniform Manifold Approximation and Proj.													•
VQ	Vector Quantization	•						•						
Total		12	6	7	14	9	9	19	14	8	6	4	28	44

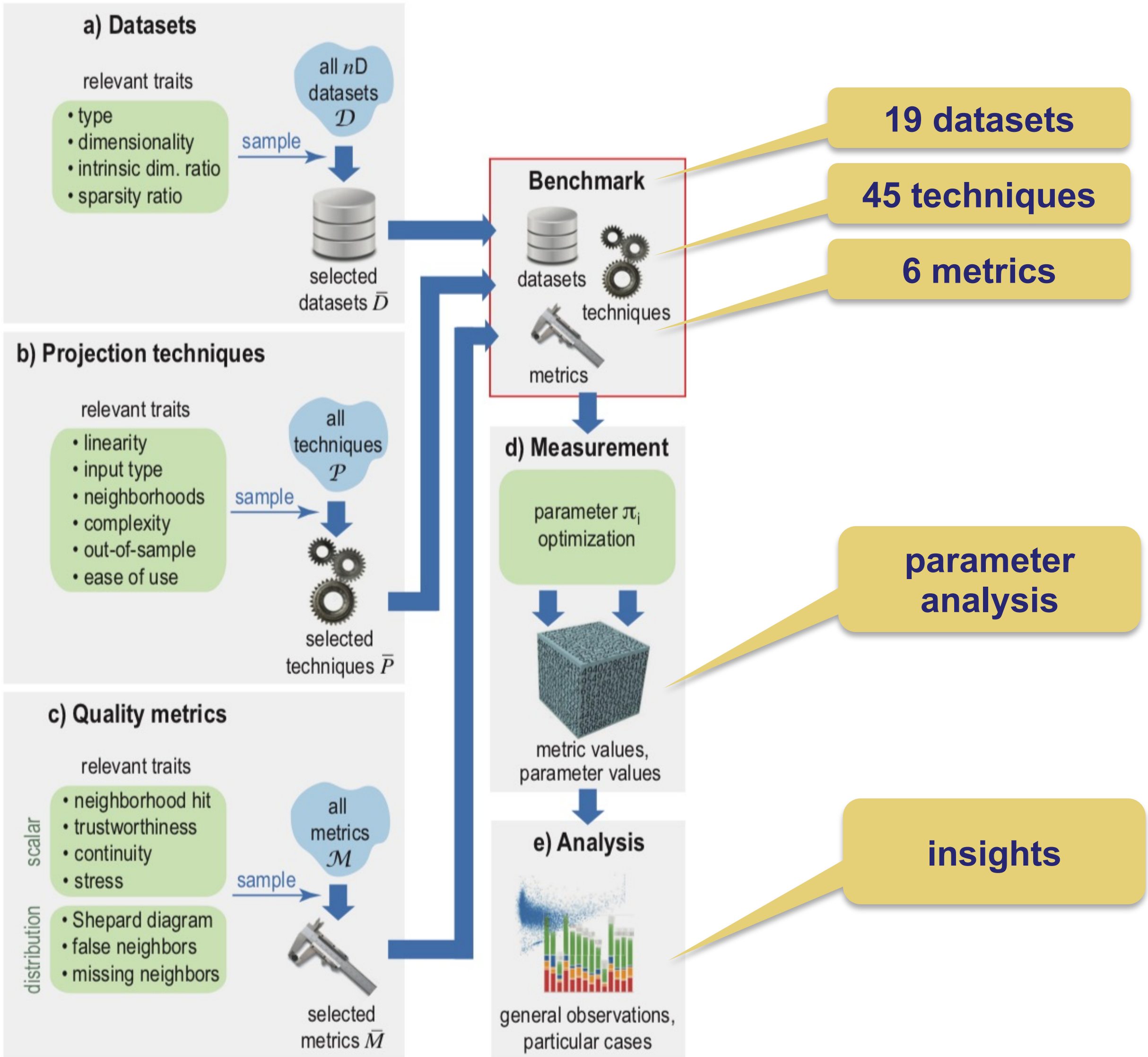
Big and unclear 'choice space'

- 50+ techniques
- 12 main surveys
- mainly theoretical discussion
- many parameters
- very limited practical comparison

Practitioner questions

- which projection is **best** for **my** context (requirements, data, …)?
- how to set its **parameters**?
- how to measure its **quality**?

Let’s measure projection quality metrics big-scale!



M. Espadoto *et al* (2019) Towards a Quantitative Survey of Dimension Reduction Techniques (IEEE TVCG)

Insights (1)

How good are projections, for which data?

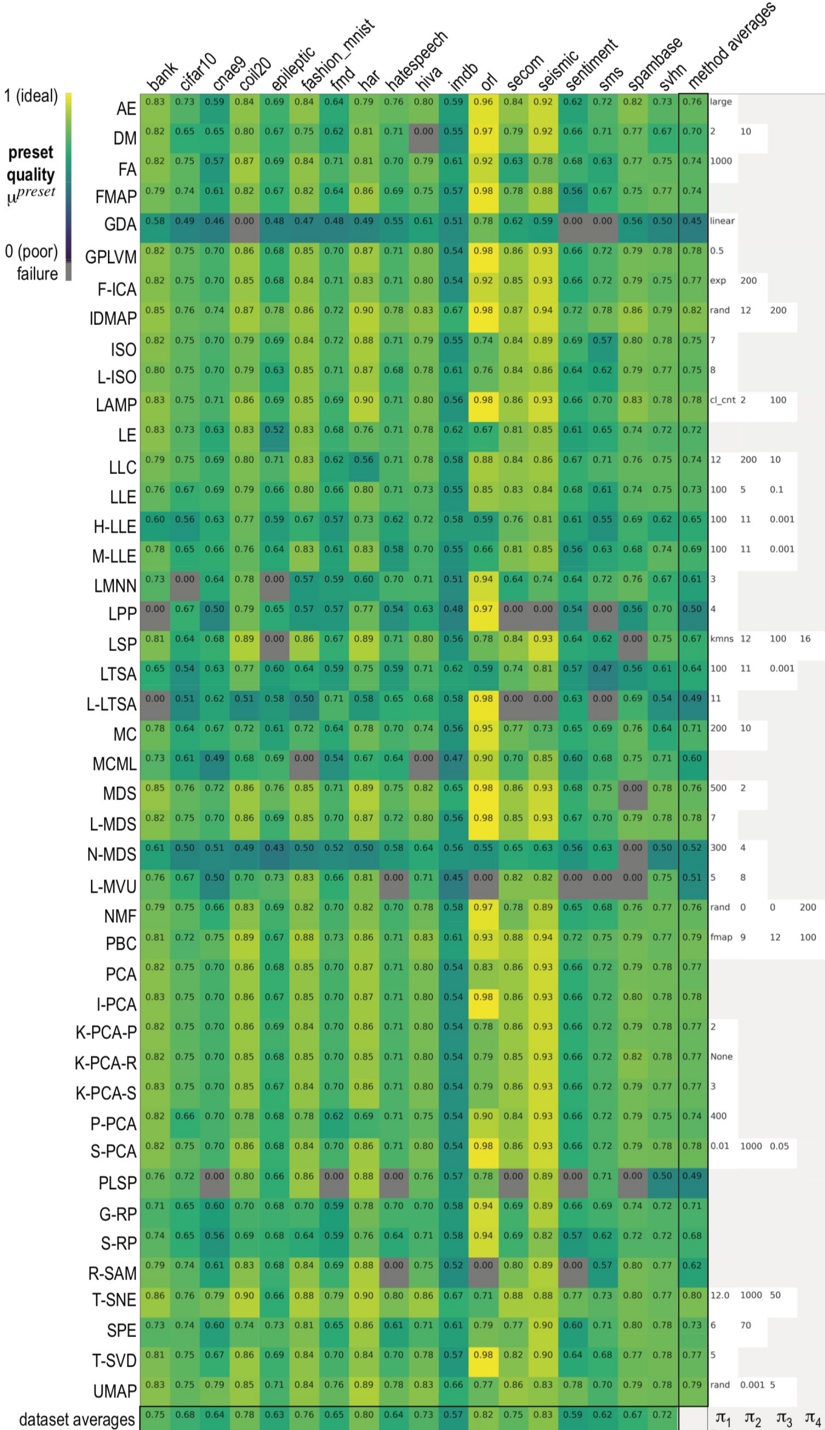
for each projection P_i
for each dataset D_j
compute *optimal* quality μ_{ij} (param. grid search)

How easy is to get optimal quality?

for each projection P_i
compute *variance* of params π_i yielding optimal quality over all datasets D_j

What we see

- no projection **best** for all dataset types
- some are quite **poor** in general (N-MDS, GDA)
- dataset **type** strongly influences quality (*imdb*: hard; *orl*: easy)
- hard to **tune** parameters to get optimal quality (large variance of π_i)



Insights (2)

How good are parameter-preset projections?

for each projection P_i

π_i^{pre} = param values yielding most times optimal quality over all datasets D_j

for each projection P_i

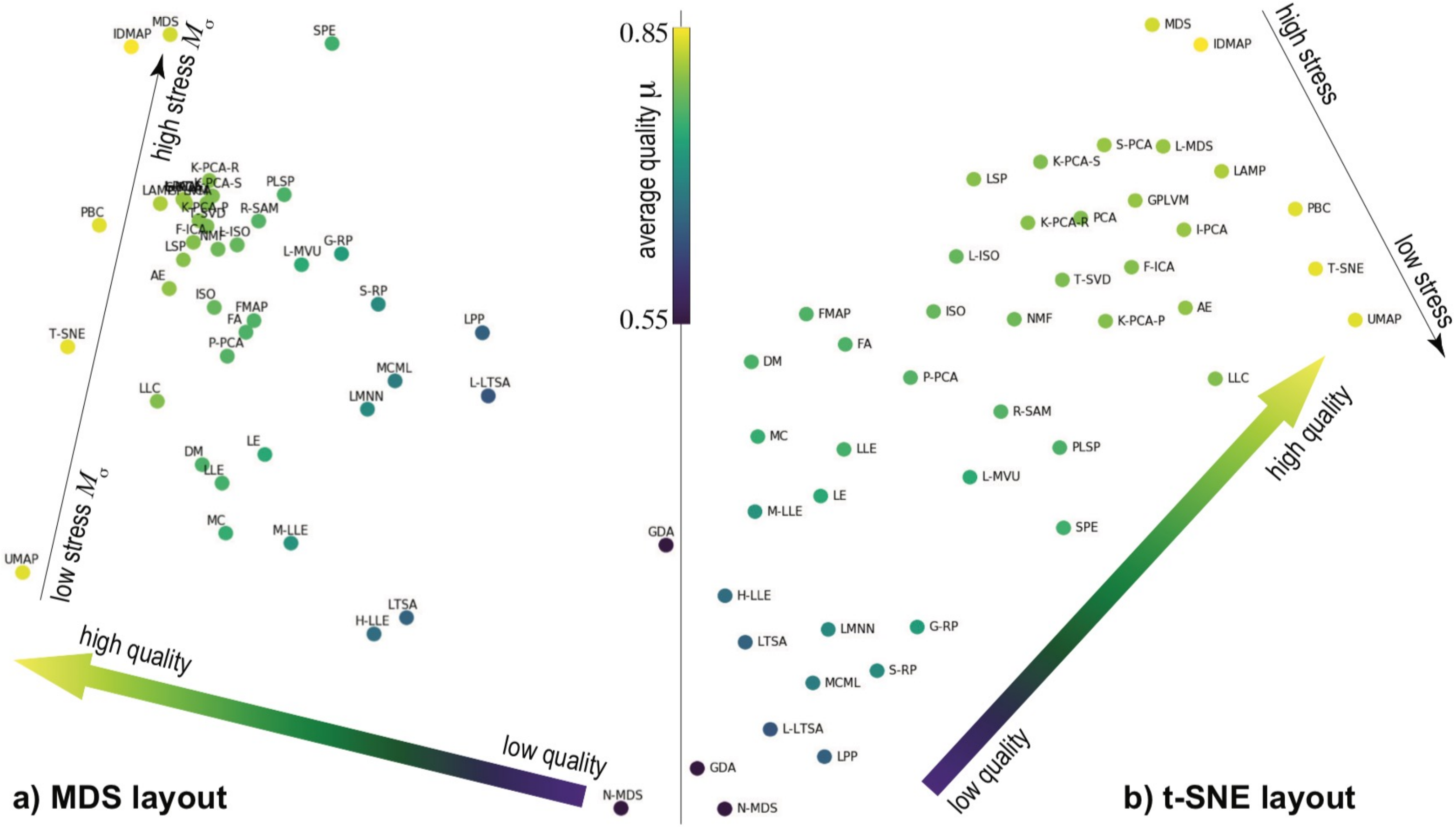
for each dataset D_j

compute quality μ_{ij} using π_i^{pre}

What we see

- very similar image to earlier one (optimal techniques stay good when using **presets**)
- again, quality strongly depends on dataset type
- t-SNE, UMAP, IDMAP, PBC score **best** on average

Insights (3): Which projections perform similarly?



'Projection of projections' map

- one point = one technique
- 5 attributes (trustworthiness, continuity, norm. stress, neighborhood hit, Shepard goodness; averaged over all tested datasets)
- we see a clear **quality trend**
- helps choosing projections that behave similarly to a user-chosen one

Benchmark

Towards A Quantitative Survey of Dimension Reduction Techniques

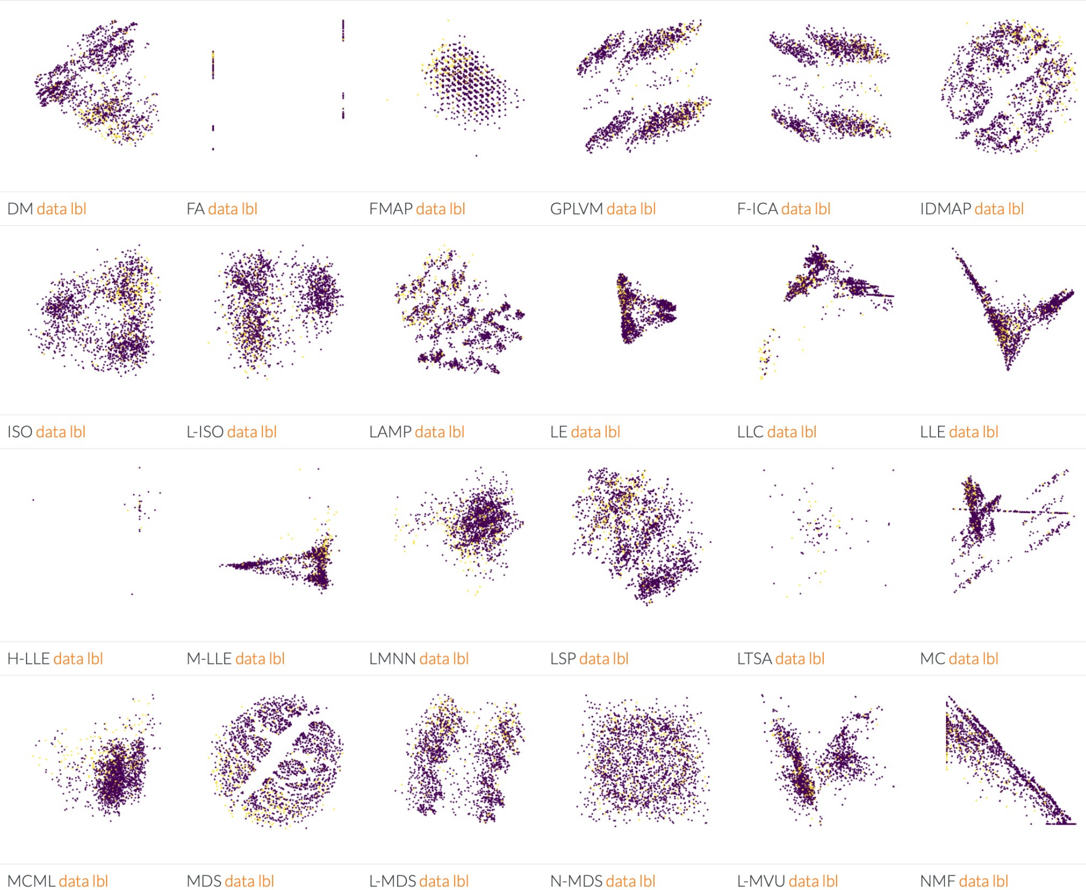
MATEUS ESPADOTO, RAFAEL M. MARTINS, ANDREAS KERREN, NINA S. T. HIRATA AND ALEXANDRU C. TELEA

DATASETS EXPERIMENT MEASUREMENTS PROJECTIONS

Projections for all datasets (best parameter set for each projection)

All projections, in csv format

bank dataset



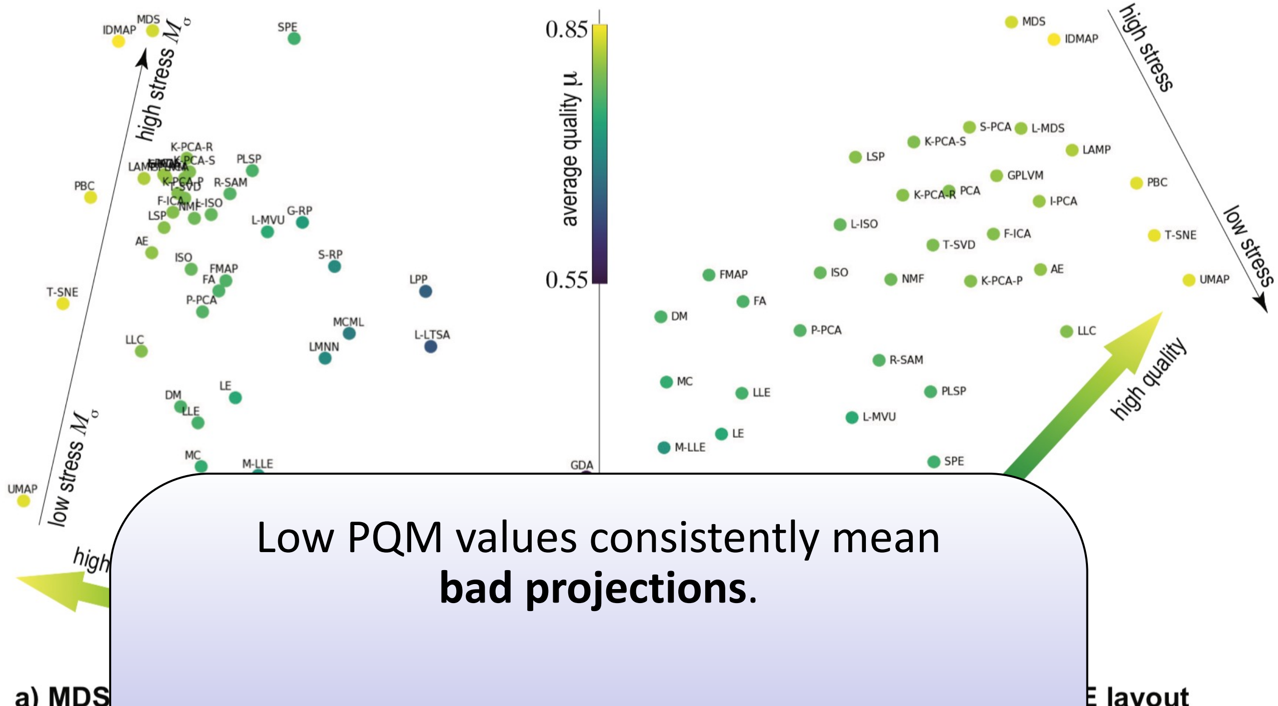
https://mespadoto.github.io/proj-quant-eval

All open source

- projection implementations
- datasets
- metric engines
- visualization engines
- optimization engines
- test harness
- all Python code

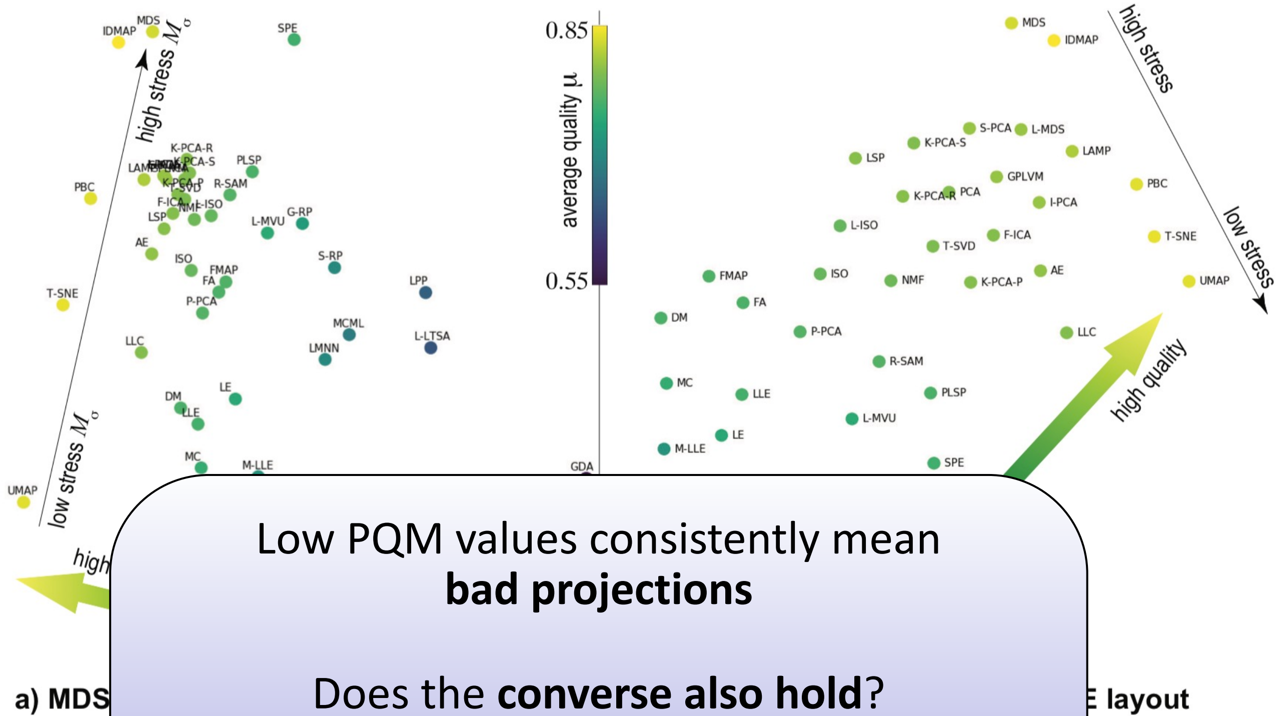
Please share, use, and extend!

Let’s recap our results



Low PQM values consistently mean **bad projections**.

Let’s recap our results

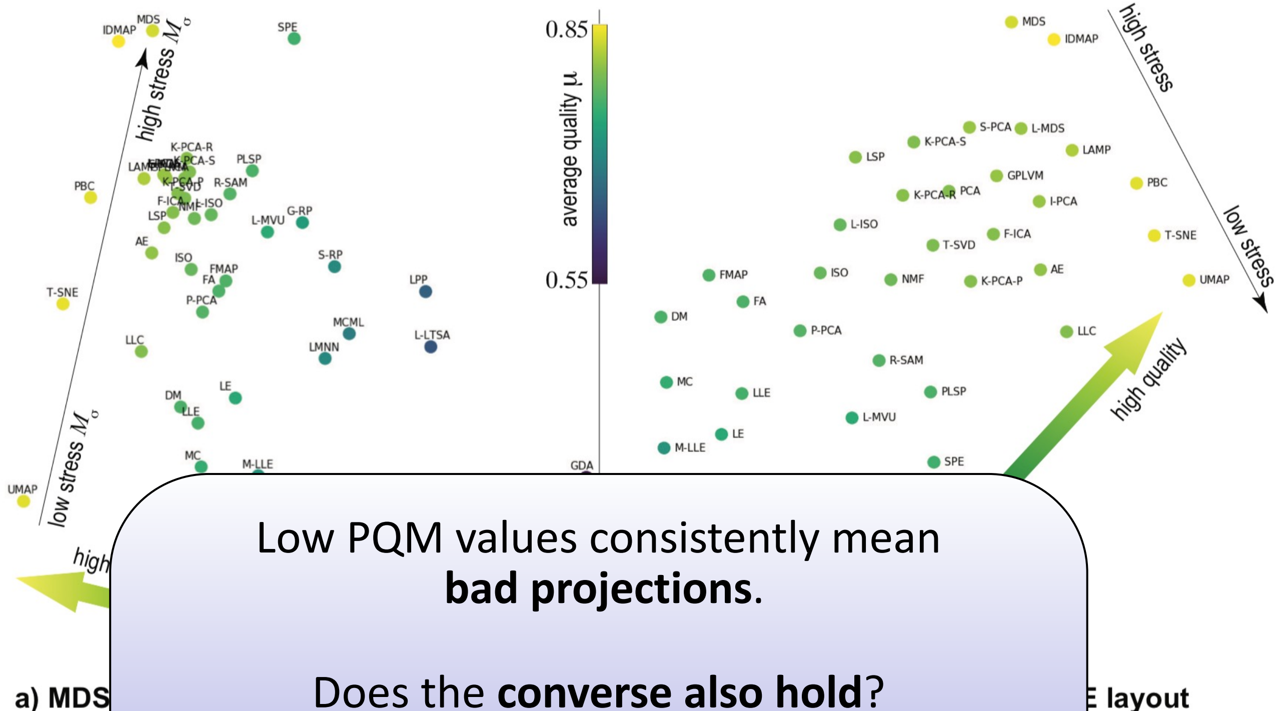


Low PQM values consistently mean
bad projections

Does the **converse also hold**?

High PQMs $\overset{?}{\Rightarrow}$ good projections?

Let’s recap our results



Low PQM values consistently mean **bad projections**.

Does the **converse also hold**?

High PQMs $\overset{?}{\Rightarrow}$ good projections?

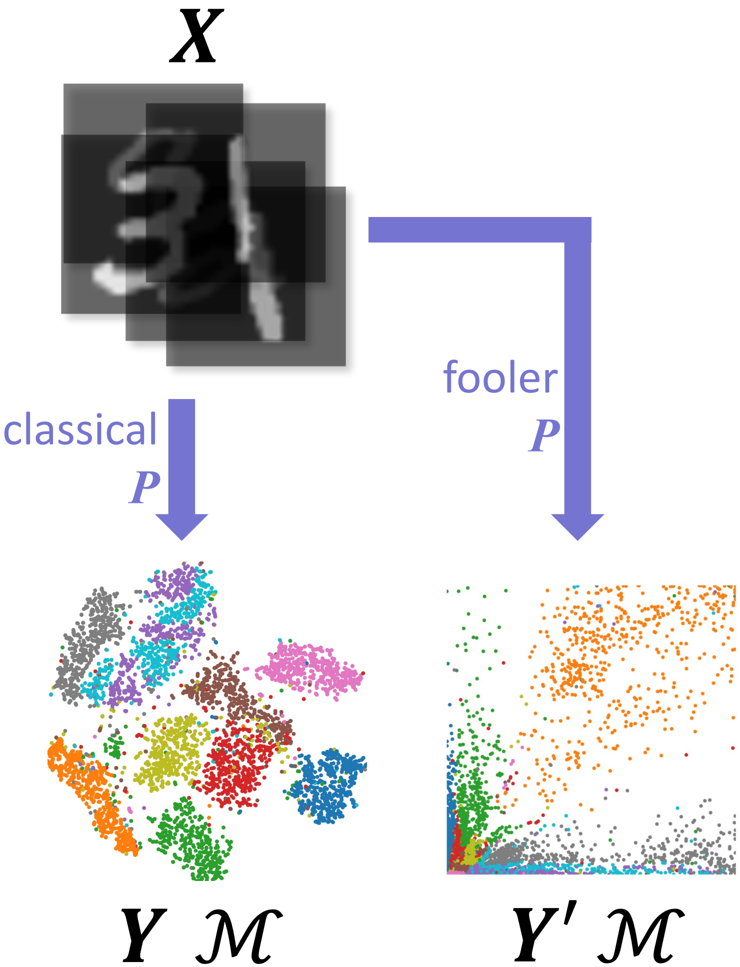
We’ll show it does not!

So PQMs are **not enough** to characterize a projection

Fooling Projection Quality Metrics

A crazy experiment follows…

Fooling Projection Quality Metrics

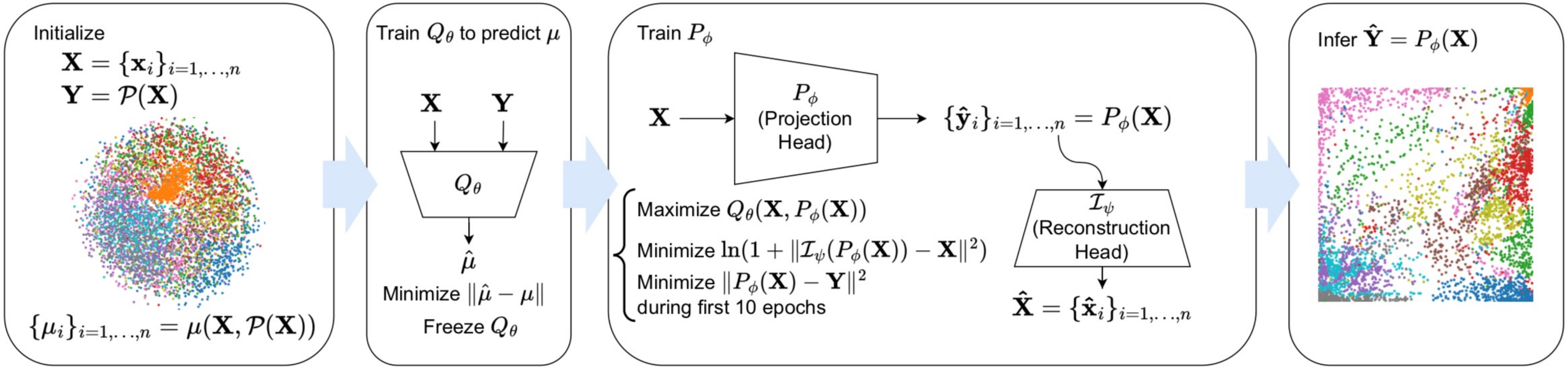


We have:
- dataset X
- projection Y
- a metric $\mathcal{M}$ having high values

We want:
- a new projection Y' of the same X
- with the same high values for $\mathcal{M}$
- poor data pattern preservation

$\Rightarrow$ $\mathcal{M}$ is not *sufficient* to identify a "good" projection

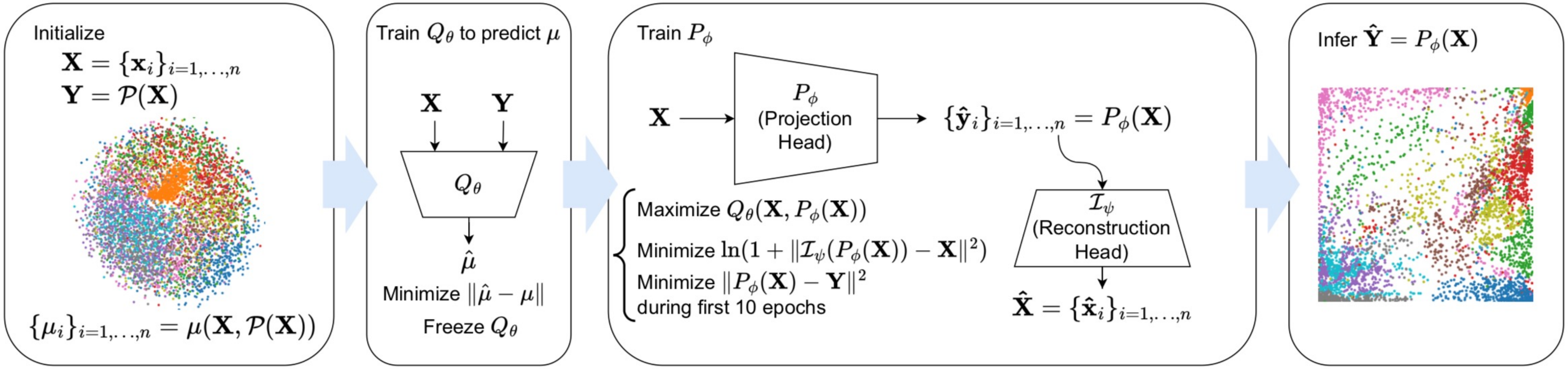
Fooling Algorithm



We start with

- some dataset X
- its projection Y
- a computed quality metric μ

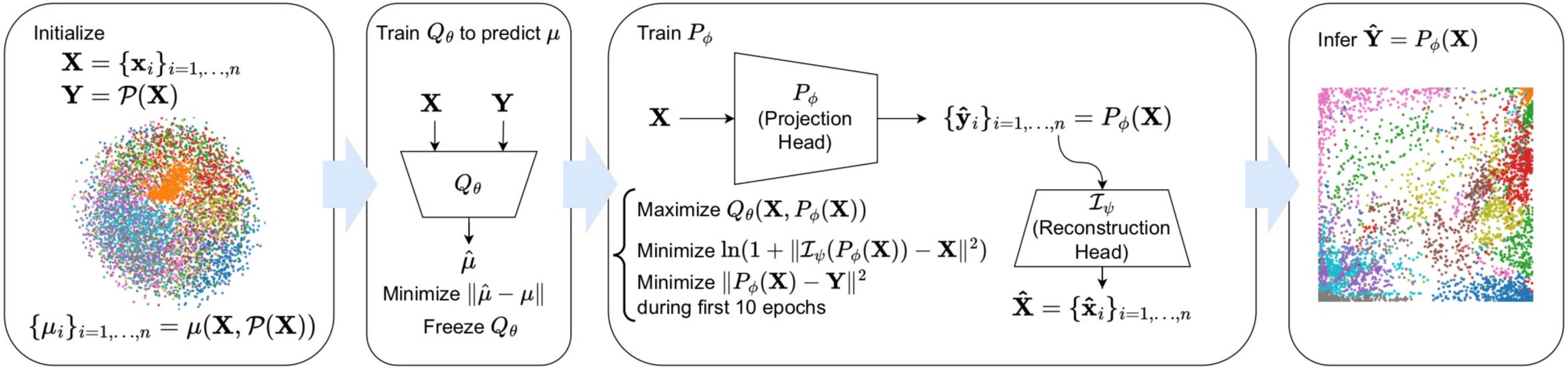
Fooling Algorithm



Train a network Q_θ to mimic the metric μ

We'll need that in **the next step**

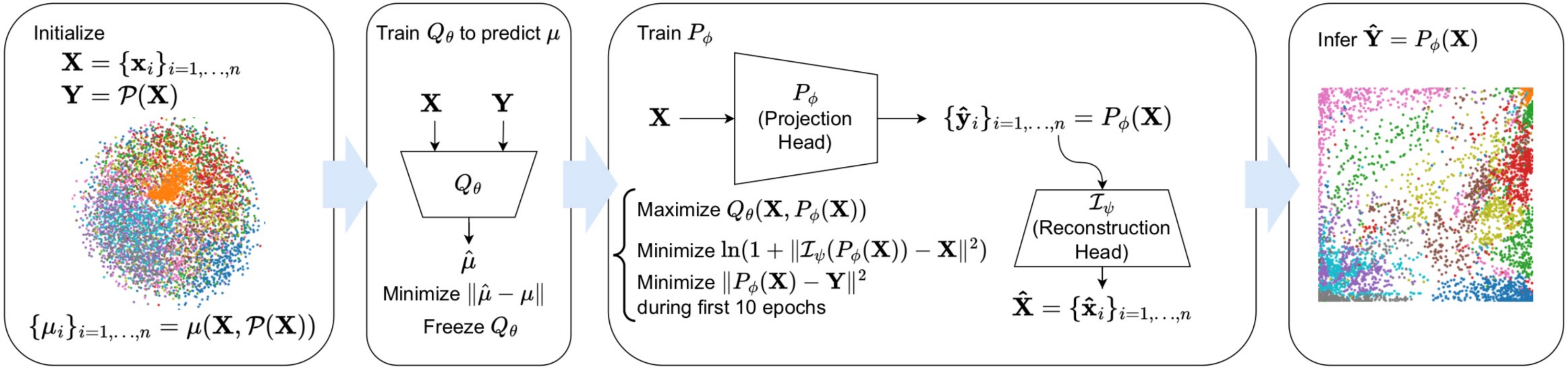
Fooling Algorithm



Train another network P_ϕ to mimic $\mathbf{Y}$
while **maximizing** μ for 10 epochs

Then, train P_ϕ to **only** maximize μ (given by Q_θ)
This allows P_ϕ to **mess up** the projection

Fooling Algorithm



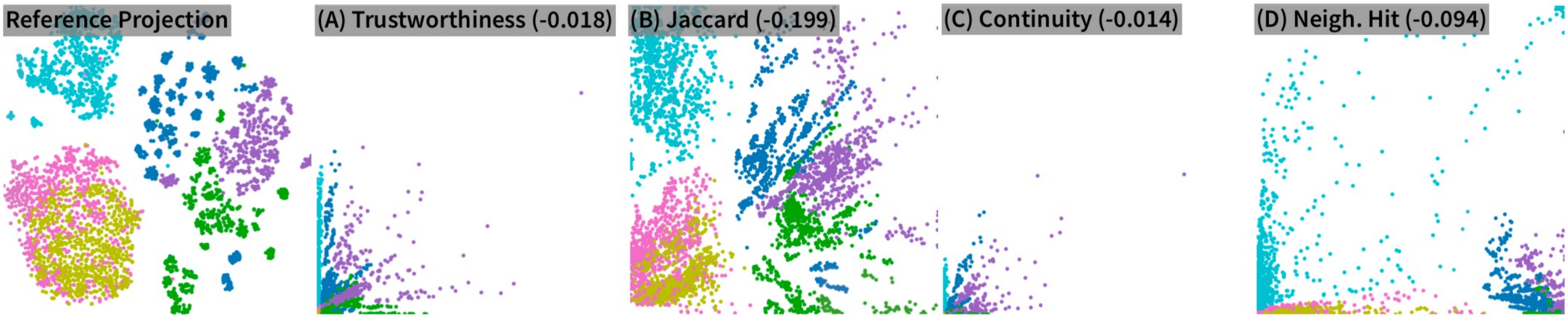
Create our fooled projection $P_\phi(\mathbf{X})$

Testing our Fooling

- 6 datasets (FashionMNIST, MNIST, HAR, Reuters, Spambase, USPS)
- 4 projection methods (t-SNE, UMAP, MDS, Isomap)
- 4 target metrics (+ all together)
- 4 parameter settings for each metric
- 17 metrics computed for each output

Metric	Parameters
Average Local Error	—
Continuity and Trustworthiness	k
Class-Aware Continuity and Trustworthiness	k
Distance Consistency (DSC)	—
Proportion of False (resp. True) Neighbors	k
Jaccard Similarity of Neighbor Sets	k
Mean Relative Ranking Errors	k
Neighborhood Hit	k
Normalized Stress	—
Pearson Correlation of Distances	—
Procrustes Statistic	k
Scale-Normalized Stress	—
Shepard Goodness	—

Results



Reference projection (HAR, t-SNE)

Fooling Trustworthiness

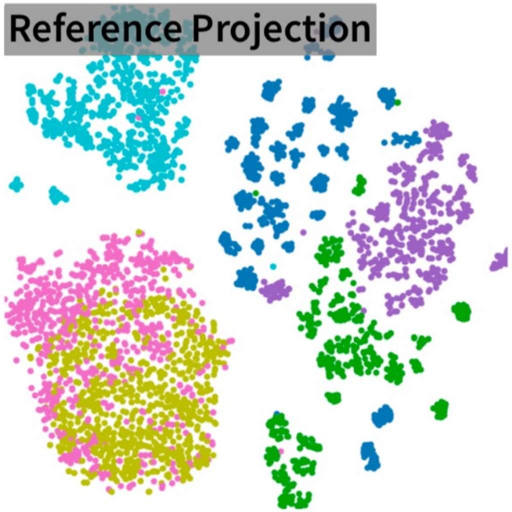
Fooling Jaccard

Fooling Continuity

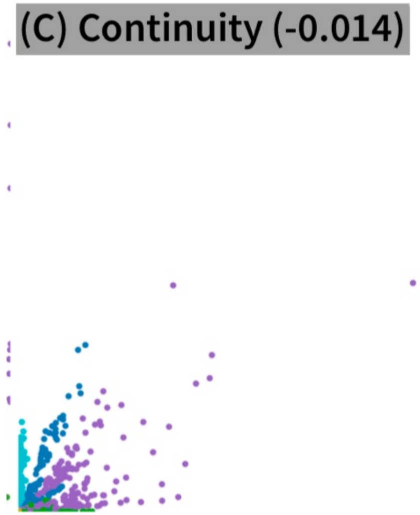
Fooling Neighborhood Hit

We lose some quality but we get completely *meaningless* projections!

Messing it up even further (in a subtle way)



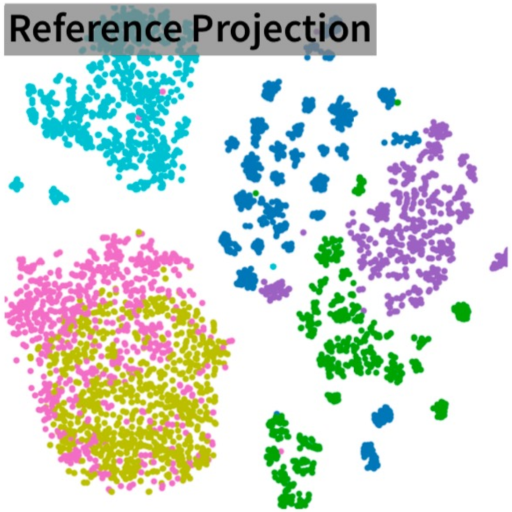
Reference
projection
(HAR, t-SNE)

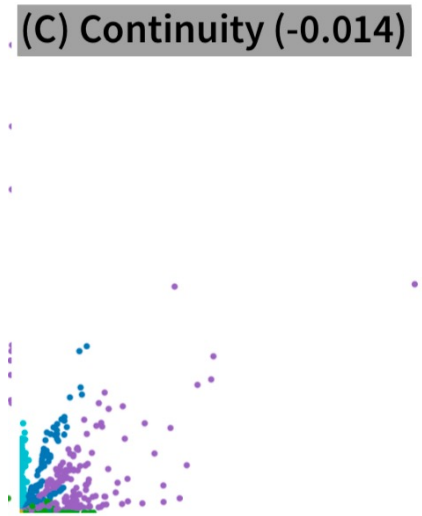


Fooling
Continuity

One will say: Sure, the quality of the right image is high but I **am not fooled** by that. It looks too **unnatural**!

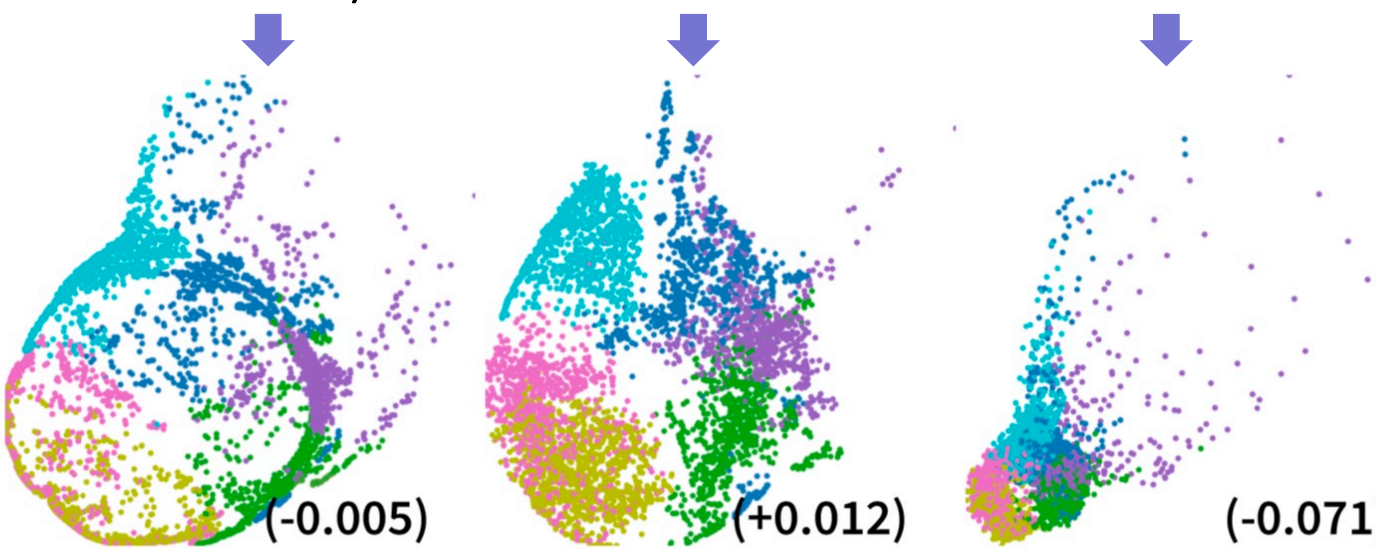
Messing it up even further (in a subtle way)





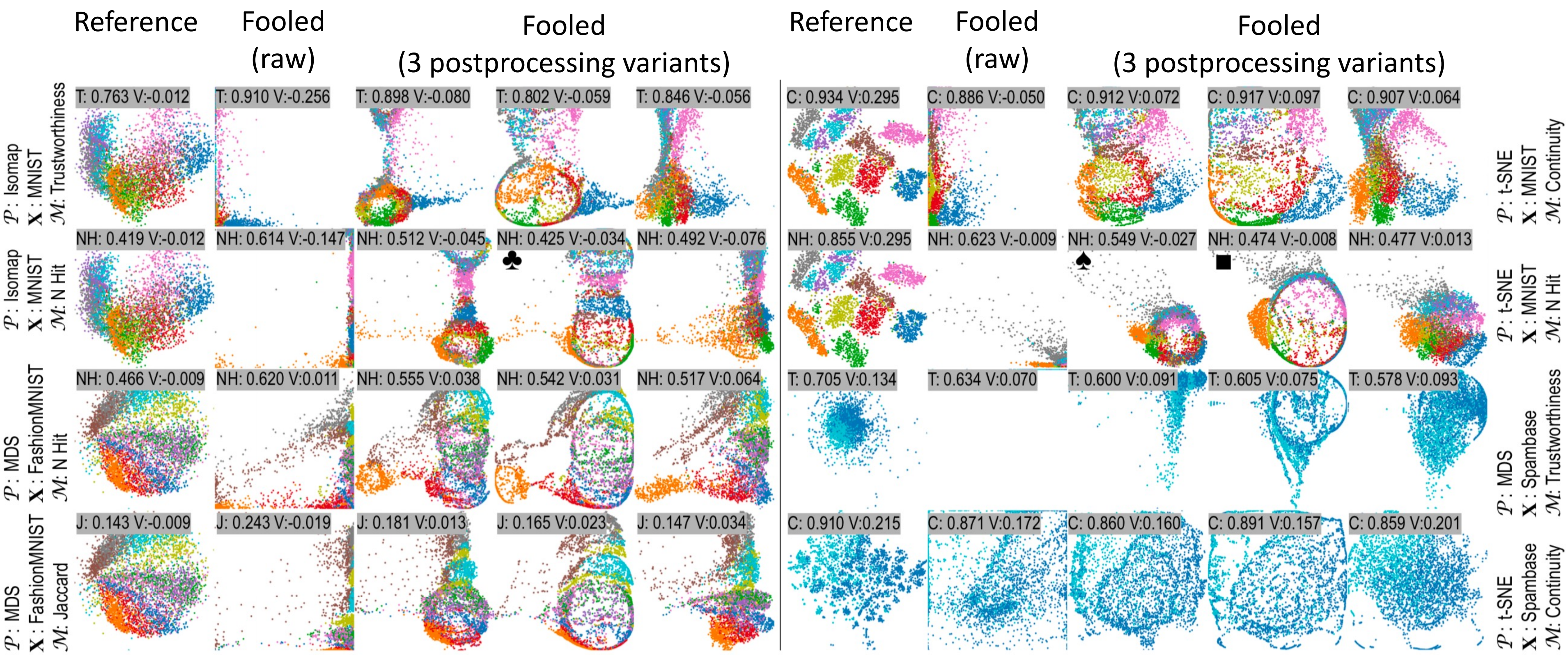
Reference projection (HAR, t-SNE)

Fooling Continuity



Use postprocessing to create **natural-looking patterns** in the fooling

More results

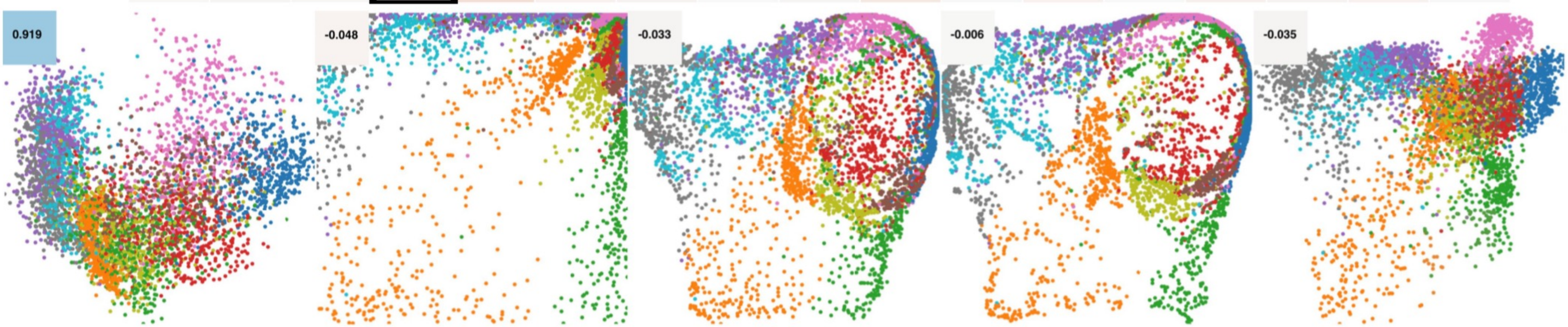


Learning to fool a metric messes up other metrics too!

MNIST

CONTINUITY

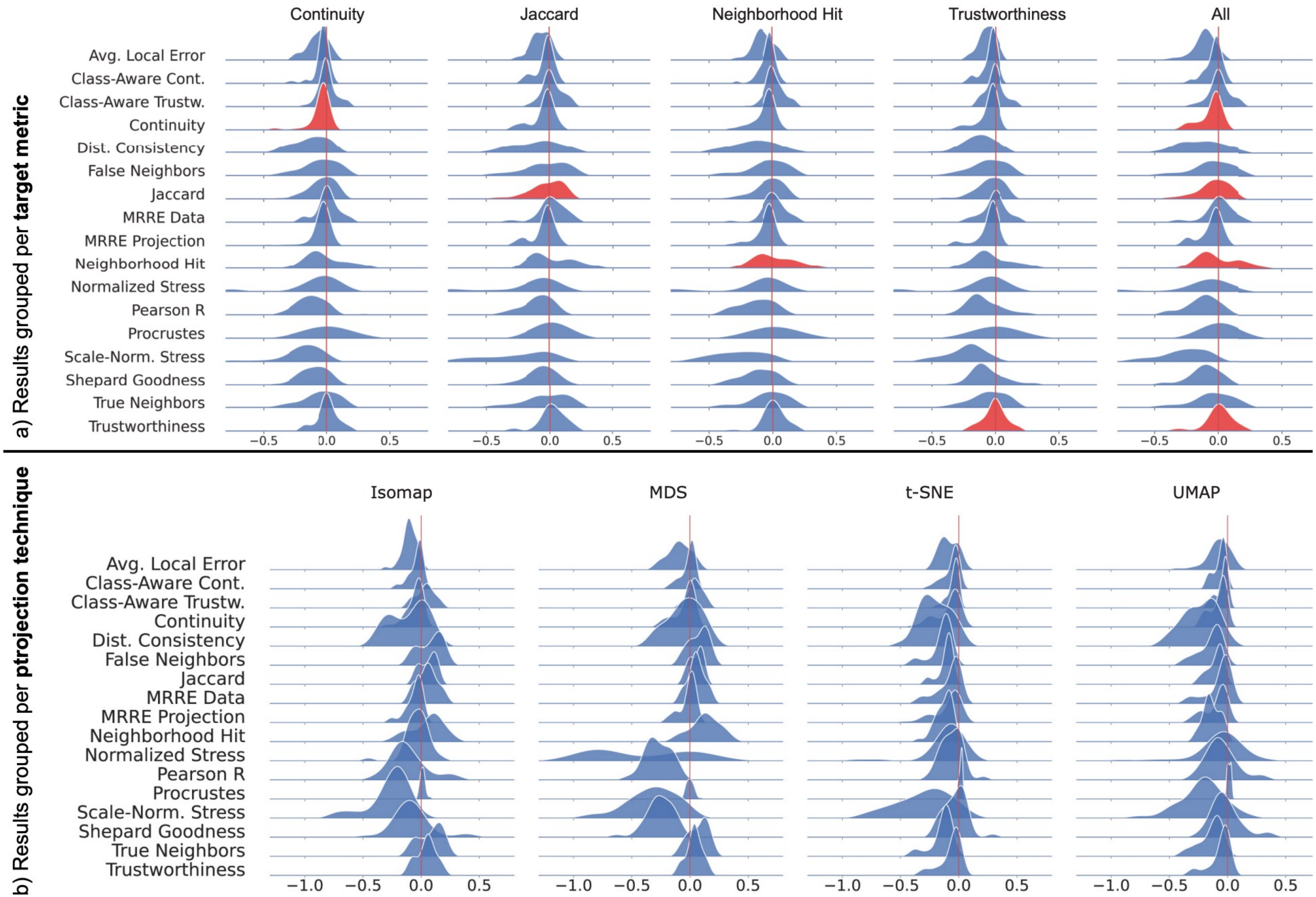
	Avg. Local Error	Class-Aware Continuity	Class-Aware Trustworthiness	Continuity	Distance Consistency	False Neighbors	Jaccard	MRRE Data	MRRE Projection	Neighborhood Hit	Normalized Stress	Pearson R	Procrustes	Scale-Normalized Stress	Shepard Goodness	True Neighbors	Trustworthiness
Isomap	**+0.024** (0.281)	**-0.017** (0.955)	**+0.155** (0.801)	**-0.048** (0.919)	**+0.024** (0.494)	**-0.192** (0.849)	**+0.133** (0.086)	**-0.190** (0.240)	**+0.046** (0.056)	**+0.238** (0.419)	**-0.010** (0.929)	**-0.441** (0.537)	**-0.003** (0.990)	**+0.193** (0.174)	**-0.443** (0.528)	**+0.192** (0.151)	**+0.170** (0.763)
MDS	**-0.055** (0.270)	**+0.032** (0.888)	**+0.120** (0.796)	**+0.020** (0.850)	**+0.053** (0.312)	**-0.127** (0.853)	**+0.084** (0.085)	**-0.147** (0.226)	**-0.032** (0.140)	**+0.234** (0.331)	**+0.726** (0.132)	**-0.287** (0.629)	**+0.024** (0.932)	**+0.068** (0.132)	**-0.277** (0.621)	**+0.127** (0.147)	**+0.113** (0.775)
t-SNE	**+0.032** (0.244)	**-0.040** (0.985)	**-0.047** (0.984)	**-0.048** (0.934)	**-0.348** (0.804)	**+0.146** (0.556)	**-0.116** (0.301)	**+0.032** (0.027)	**+0.049** (0.040)	**-0.245** (0.855)	**+0.001** (0.915)	**-0.150** (0.422)	**-0.014** (0.988)	**+0.151** (0.158)	**-0.105** (0.393)	**-0.146** (0.444)	**-0.042** (0.951)
UMAP	**+0.014** (0.244)	**-0.036** (0.988)	**-0.028** (0.983)	**-0.031** (0.931)	**-0.302** (0.845)	**+0.072** (0.586)	**-0.056** (0.274)	**+0.000** (0.051)	**+0.039** (0.041)	**-0.185** (0.854)	**-0.012** (0.907)	**-0.081** (0.400)	**-0.006** (0.988)	**+0.096** (0.172)	**-0.066** (0.367)	**-0.072** (0.414)	**-0.011** (0.940)



Reference | Fooled (raw) | Fooled (3 postprocessing variants)

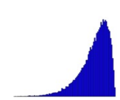
Check it up yourself: https://amreis.github.io/fool-proj-metrics/

How well can we fool *all* metrics?

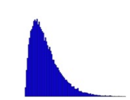


Plotted values: Distributions of $\mathcal{M}(\boldsymbol{Y}')$ - $\mathcal{M}(\boldsymbol{Y})$

Red: metric we aim to fool

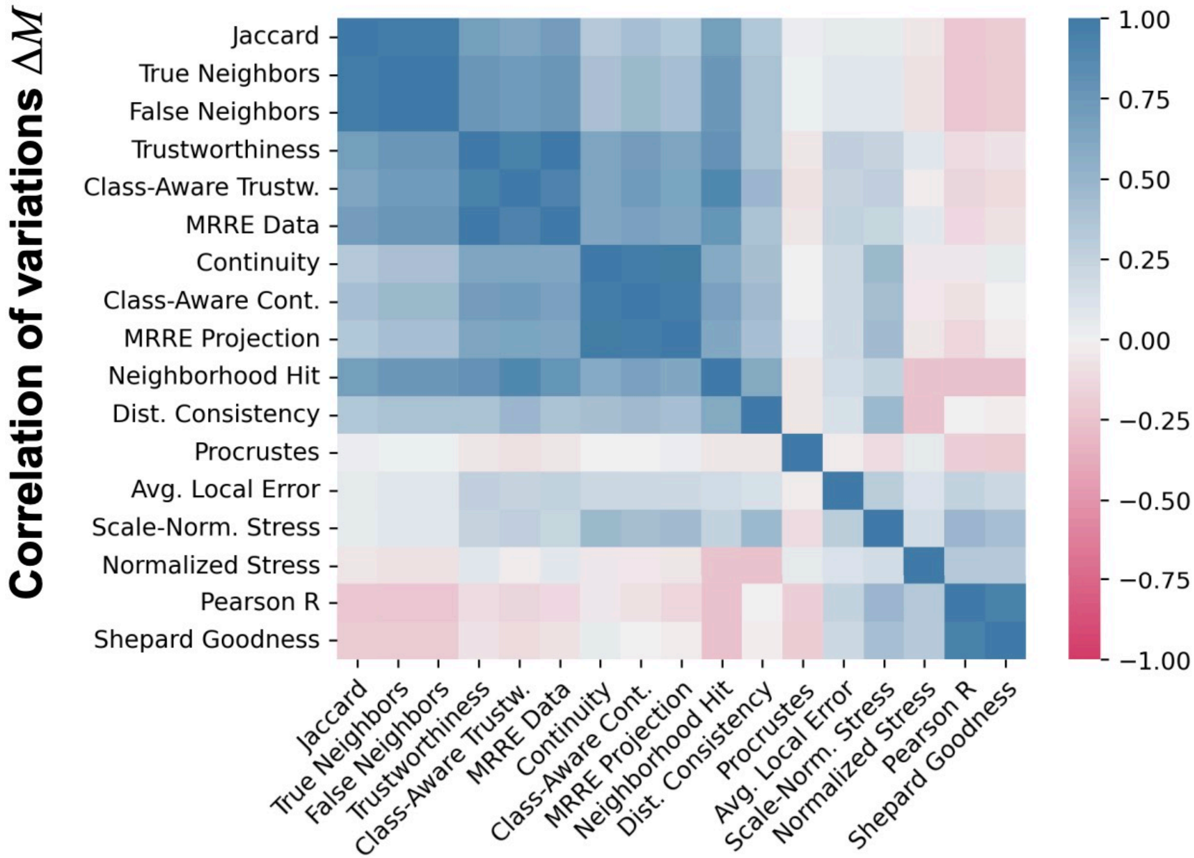


fooling increases quality



fooling decreases quality

How does fooling a metric affect other metrics?



Blue cells: Metrics that can be fooled ‘together’
(train to fool one metric fools also the others)

⇒ **uncorrelated** metrics are most interesting to study

Which metrics to use?

Cluster all 17 metrics on **correlation**
⇒ correlated metrics get in same cluster

Pick **one metric** in each cluster (plus the unclustered ones)

1

- Avg. Local Error
- Pearson Corr. of Distances ⭐
- Shepard Goodness
- Scale-Normalized Stress

2

- Jaccard
- % False Neighbors
- % True Neighbors ⭐

4

- Class-Aware Trustworthiness
- Trustworthiness
- MRRE Data

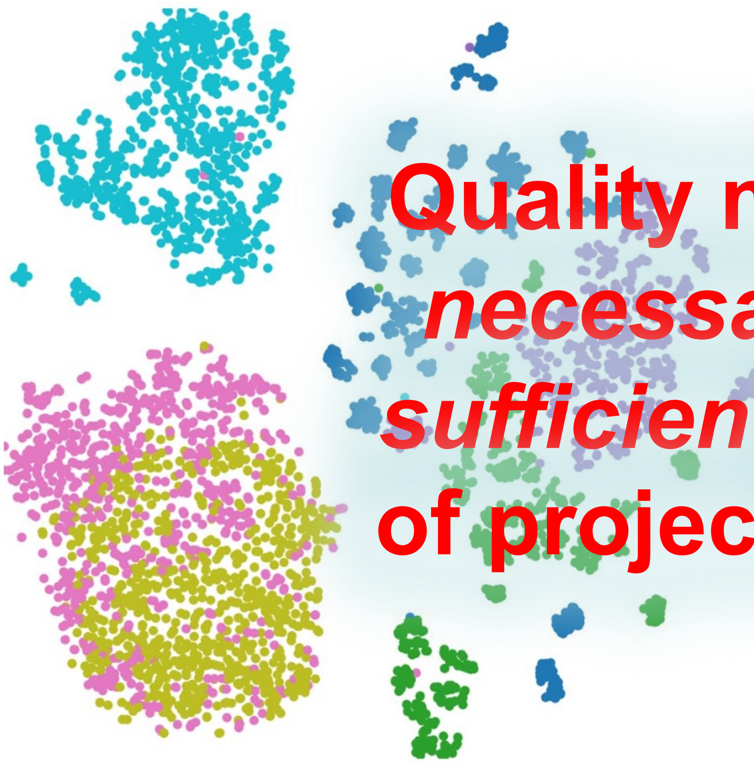
3

- Class-Aware Continuity
- Continuity
- MRRE Proj.

- Distance Consistency ⭐
- Neighborhood Hit ⭐
- Normalized Stress
- Procrustes Statistic

The Main Takeaway

HAR (t-SNE projection)



T: 0.99 C: 0.99 NH: 0.94 Jac: 0.37

Our fooled projection

T: 0.98 C: 0.97 NH: 0.87 Jac: 0.13

Quality metrics are *necessary* but not *sufficient* indicators of projection quality

Thanks go to my team

webspace.science.uu.nl/~telea001

a.c.telea@uu.nl

vig.science.uu.nl