

Package: manifoldsculpting (via r-universe)

October 27, 2026

Type Package

Title Nonlinear Dimensionality Reduction by Manifold Sculpting

Version 1.2.1

Description Manifold Sculpting (Gashler, Ventura and Martinez (2008)

<<https://proceedings.neurips.cc/paper/2007/hash/c06d06da9666a219db15cf575aff2824-Abstract.html>>)
is an iterative nonlinear dimensionality reduction method. It records the distances and angles between each point and its nearest neighbors, then gradually scales the data out of the dimensions to be removed while moving the points in the remaining dimensions to restore these local relationships, as if surface tension were acting on the manifold. New samples are embedded by solving the same local problem against the training embedding. The neighbor search and the sculpting iterations are written in C++ with 'RcppArmadillo', and step_manifold_sculpting() makes the method available as a 'recipes' preprocessing step for 'tidymodels' workflows.

License MIT + file LICENSE

URL <https://github.com/ChristianGoueguel/manifold-sculpting>,
<https://christiangoueguel.com/manifold-sculpting/>

BugReports <https://github.com/ChristianGoueguel/manifold-sculpting/issues>

Depends R (>= 4.1.0), recipes (>= 1.1.0)

Imports cli, dials (>= 1.2.0), generics, Rcpp, rlang (>= 1.1.0), stats, tibble, vctrs, withr

Suggests ggplot2, knitr, parsnip, rmarkdown, testthat (>= 3.0.0), workflows

LinkingTo Rcpp, RcppArmadillo

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)
Config/roxygen2/version 8.1.0
Config/pak/sysreqs libicu-dev
Repository https://christiangoueguel.r-universe.dev
Date/Publication 2026-10-27 12:47:41 UTC
RemoteUrl https://github.com/ChristianGoueguel/manifold-sculpting
RemoteRef HEAD
RemoteSha e2905c24fae98494c3c828024732e0d4807d6319

Contents

manifold_sculpting	2
predict.manifold_sculpting	5
sim_manifolds	6
squishing_rate	7
step_manifold_sculpting	7
Index	10

manifold_sculpting	<i>Manifold Sculpting</i>
--------------------	---------------------------

Description

manifold_sculpting() reduces the dimension of numeric data with Manifold Sculpting (Gashler, Ventura and Martinez, 2008), an iterative nonlinear dimensionality reduction method that preserves the distances and angles between neighboring points.

Usage

```
manifold_sculpting(  
  x,  
  ndim = 2,  
  neighbors = 20,  
  squishing_rate = 0.99,  
  nstart = 1,  
  max_iter = 5000,  
  patience = 50,  
  init = "pca",  
  nn_method = c("auto", "kd_tree", "brute_force"),  
  verbose = FALSE  
)
```

Arguments

x	A numeric matrix or data frame, with one row per sample. Distances are Euclidean, so variables on different scales should be standardized first.
ndim	The number of dimensions of the embedding. It must be smaller than the number of columns of x.
neighbors	The number of nearest neighbors, k, that define the local relationships to preserve. With too few neighbors, the embedding is more likely to fold onto itself; neighborhoods that are too large cut across the folds of the manifold.
squishing_rate	The factor, sigma in (0, 1), by which the dimensions to remove are scaled down at each iteration. Values closer to 1 (0.99 in the paper) are slower but follow the manifold more closely.
nstart	The number of runs. The runs start from the same configuration but take different random paths, and the embedding with the lowest final error is returned. Several runs make the result more reliable on manifolds that are hard to unfold (see Details).
max_iter	The maximum number of iterations.
patience	Once the dimensions to remove have shrunk below 1% of their size, the iterations stop when the error has not improved for patience iterations.
init	The starting configuration. "pca" (the default) rotates the data onto their principal components, as recommended in the paper. Alternatively, a numeric matrix with one row per sample and at least ndim columns, such as the result of another dimensionality reduction method: its first ndim columns are refined, and its other columns, if any, are scaled down.
nn_method	The nearest neighbor search: "kd_tree", "brute_force" (computed with BLAS), or "auto", which uses a k-d tree for data with up to 10 columns. Both are exact.
verbose	Print the progress every 50 iterations?

Details

The algorithm has five steps:

1. Find the k nearest neighbors of each point.
2. For each point p_i and neighbor n_{ij} , record the distance δ_{ij} between them, and the angle θ_{ij} formed by p_i , n_{ij} and m_{ij} , the neighbor of n_{ij} that is the most colinear with p_i . The average neighbor distance is δ_{ave} .
3. Rotate the data onto their principal components. The first ndim components are the preserved dimensions; the others are scaled.
4. Iterate: multiply the scaled dimensions by squishing_rate, then scale the preserved dimensions up until the average neighbor distance is δ_{ave} again. Visit the points breadth-first, from a random point, and move each one in the preserved dimensions, by hill climbing, to minimize the error

$$\epsilon_i = \sum_j w_{ij} \left[\left(\frac{\delta_{ij} - \delta_{ij}^0}{\delta_{ave}} \right)^2 + \max \left(0, \frac{\theta_{ij}^0 - \theta_{ij}}{\pi} \right)^2 \right]$$

where the superscript 0 marks the original values, and the weights w_{ij} are higher for the neighbors that have already been moved in this iteration.

5. Drop the scaled dimensions.

The weighting and the normalization of the error, the adaptive step size of the hill climbing, and the stopping rule follow the reference implementation of the authors in the Waffles library. Unlike Waffles, all the dimensions of the data are kept (as the paper asks): the scaled dimensions only contribute to the distances and angles through a common factor, so their contributions are computed once and the iterations only work on the `ndim` preserved dimensions. Points that the breadth-first traversal cannot reach are also adjusted at each iteration.

Like any local optimization, the iterations can end in a local minimum, where part of the manifold is folded over another. The final error tells these runs apart: it is much larger than for a run that unfolded the manifold. The risk depends on the data, and decreases with more neighbors and more runs (`nstart`). On 2000 points of the Swiss roll of `sim_swiss_roll()`, for example, about one run in seven stays folded with 20 neighbors (more with fewer points), but the best of three runs is reliable, while the S-curve of `sim_s_curve()` is unfolded with 14 neighbors.

The neighbor search and the iterations are written in C++. The algorithm uses the random number generator of R; call `set.seed()` first for reproducible results.

Value

An object of class `manifold_sculpting`, a list with:

embedding The embedding, a matrix with one row per sample and `ndim` columns.

error The total error at each iteration (of the best run).

iterations The number of iterations (of the best run).

converged Whether the iterations stopped before `max_iter`.

final_error The final error of each run.

and the elements needed by `predict.manifold_sculpting()` to embed new samples.

References

Gashler, M., Ventura, D. and Martinez, T. (2008). Iterative non-linear dimensionality reduction with manifold sculpting. In *Advances in Neural Information Processing Systems 20*, 513–520. <https://proceedings.neurips.cc/paper/2007/hash/c06d06da9666a219db15cf575aff2824-Abstract.html>

Gashler, M. S. (2011). Waffles: A machine learning toolkit. *Journal of Machine Learning Research*, 12, 2383–2387.

See Also

`predict.manifold_sculpting()` to embed new samples, and `step_manifold_sculpting()` for the 'recipes' step.

Examples

```
set.seed(1)
curve <- sim_s_curve(800)
fit <- manifold_sculpting(curve[c("x", "y", "z")], ndim = 2)
fit
```

```
# The embedding recovers the intrinsic coordinates of the S-curve
cor(fit$embedding, curve[c("u", "v")])
```

predict.manifold_sculpting

Embed new samples with Manifold Sculpting

Description

Manifold Sculpting has no explicit mapping from the data to the embedding. A new sample is embedded from its neighbors nearest training samples: it starts at the combination of their embeddings given by its locally linear reconstruction weights (as in locally linear embedding), and is then moved to minimize the Manifold Sculpting error, the discrepancy between its original and embedded distances and angles to its neighbors, whose embeddings are held fixed. A new sample that is identical to a training sample takes its embedding, so the training data are mapped to `manifold_sculpting()`'s embedding.

Usage

```
## S3 method for class 'manifold_sculpting'
predict(object, new_data, refine = TRUE, ...)
```

Arguments

<code>object</code>	A <code>manifold_sculpting()</code> fit.
<code>new_data</code>	A numeric matrix or data frame with the variables used to fit <code>object</code> . Columns are matched by name when <code>object</code> was fitted with column names, and by position otherwise.
<code>refine</code>	If <code>FALSE</code> , the samples stay at their locally linear reconstruction, without the error minimization.
<code>...</code>	Not used.

Value

A matrix with one row per sample of `new_data` and `object$ndim` columns.

Examples

```
set.seed(1)
roll <- sim_swiss_roll(1000)
train <- roll[1:800, ]
test <- roll[801:1000, ]
fit <- manifold_sculpting(train[c("x", "y", "z")], nstart = 3)
head(predict(fit, test))
```

sim_manifolds

Simulated manifolds

Description

The Swiss roll and S-curve manifolds of Gashler, Ventura and Martinez (2008), with their intrinsic (two-dimensional) coordinates, to test dimensionality reduction methods.

Usage

```
sim_swiss_roll(n = 2000, noise = 0)
```

```
sim_s_curve(n = 2000, noise = 0)
```

Arguments

n	The number of points.
noise	The standard deviation of Gaussian noise added to the coordinates x, y and z.

Details

For the Swiss roll, $t_i = 8i/n + 2$ for $0 \leq i < n$, and

$$x_i = t_i \sin(t_i), \quad y_i \sim U(-6, 6), \quad z_i = t_i \cos(t_i).$$

Its intrinsic coordinates are the arc length along the roll, $u_i = (\sinh^{-1}(t_i) + t_i \sqrt{t_i^2 + 1})/2$, and $v_i = y_i$.

For the S-curve, $t_i = (2.2i - 0.1)\pi/n$, and

$$x_i = t_i, \quad y_i = \sin(t_i), \quad z_i \sim U(0, 2).$$

Its intrinsic coordinates are the arc length along the curve, $u_i = \int_0^{t_i} \sqrt{\cos^2(w) + 1} dw$, and $v_i = z_i$. (The paper gives $v_i = y_i$, a misprint since y_i is a function of u_i .)

Value

A tibble with n rows and the columns x, y, z (the coordinates in three dimensions) and u, v (the intrinsic coordinates). The points are ordered by u.

References

Gashler, M., Ventura, D. and Martinez, T. (2008). Iterative non-linear dimensionality reduction with manifold sculpting. In *Advances in Neural Information Processing Systems 20*, 513–520.

Examples

```
set.seed(1)
sim_swiss_roll(5)
sim_s_curve(5)
```

squishing_rate	<i>Squishing rate parameter</i>
----------------	---------------------------------

Description

The rate at which `manifold_sculpting()` scales down the dimensions to remove, for tuning `step_manifold_sculpting()`.

Usage

```
squishing_rate(range = c(0.95, 0.995), trans = NULL)
```

Arguments

range	A two-element vector holding the defaults for the smallest and largest possible values.
trans	A trans object from the scales package, or NULL.

Value

A `quant_param` object from the 'dials' package.

Examples

```
squishing_rate()
```

step_manifold_sculpting	<i>Manifold Sculpting signal extraction</i>
-------------------------	---

Description

`step_manifold_sculpting()` creates a *specification* of a recipe step that converts numeric data into one or more new dimensions with Manifold Sculpting.

Usage

```
step_manifold_sculpting(
  recipe,
  ...,
  role = "predictor",
  trained = FALSE,
  num_comp = 2,
  neighbors = 20,
  squishing_rate = 0.99,
```

```

options = list(),
seed = sample.int(10^5, 1),
res = NULL,
columns = NULL,
prefix = "MS",
keep_original_cols = FALSE,
skip = FALSE,
id = recipes::rand_id("manifold_sculpting")
)

## S3 method for class 'step_manifold_sculpting'
tidy(x, ...)

```

Arguments

recipe	A recipe object. The step will be added to the sequence of operations for this recipe.
...	One or more selector functions to choose variables for this step. See recipes::selections() for more details.
role	For model terms created by this step, what analysis role should they be assigned? By default, the new columns created by this step from the original variables will be used as <i>predictors</i> in a model.
trained	A logical to indicate if the quantities for preprocessing have been estimated.
num_comp	The number of dimensions of the embedding. It is reduced to one less than the number of selected columns when it is larger. With <code>num_comp = 0</code> , the data are returned unchanged.
neighbors	The number of nearest neighbors that define the local relationships to preserve.
squishing_rate	The factor by which the dimensions to remove are scaled down at each iteration, in (0, 1).
options	A list of other arguments to manifold_sculpting() : <code>nstart</code> , <code>max_iter</code> , <code>patience</code> , <code>nn_method</code> and <code>verbose</code> .
seed	A single integer, the seed of the random numbers used when the step is trained, for reproducible results.
res	The manifold_sculpting() fit, once the step has been trained by recipes::prep() .
columns	A character string of the selected variable names. This field is a placeholder and will be populated once prep() is used.
prefix	The prefix of the names of the new columns, which are numbered from 1 (MS1, MS2, ...).
keep_original_cols	A logical to keep the original variables in the output. Defaults to FALSE.
skip	A logical. Should the step be skipped when the recipe is baked by bake() ? While all operations are baked when prep() is run, some operations may not be able to be conducted on new data (e.g. processing the outcome variable(s)). Care should be taken when using <code>skip = TRUE</code> as it may affect the computations for subsequent operations.

id	A character string that is unique to this step to identify it.
x	A step_manifold_sculpting object.

Details

Manifold Sculpting (Gashler, Ventura and Martinez, 2008) finds a low-dimensional embedding that preserves the distances and angles between neighboring samples; see `manifold_sculpting()` for the algorithm. It is based on Euclidean distances, so the predictors should usually be centered and scaled first, with `recipes::step_normalize()`.

New samples, given to `recipes::bake()`, are embedded from their nearest training samples by `predict.manifold_sculpting()`. The training set gets the embedding found by the algorithm.

Value

An updated version of recipe with the new step added to the sequence of any existing operations.

Tidying

When you `tidy()` this step, a tibble is returned with the columns `terms` (the selectors or variables selected) and `id`.

Tuning parameters

This step has 3 tuning parameters: `num_comp` (# Components), `neighbors` (# Nearest Neighbors) and `squishing_rate` (Squishing Rate, see `squishing_rate()`).

Case weights

The underlying operation does not allow for case weights.

References

Gashler, M., Ventura, D. and Martinez, T. (2008). Iterative non-linear dimensionality reduction with manifold sculpting. In *Advances in Neural Information Processing Systems 20*, 513–520.

Examples

```
library(recipes)

set.seed(1)
roll <- sim_swiss_roll(1000)
train <- roll[1:800, ]
test <- roll[801:1000, ]

rec <- recipe(~ x + y + z, data = train) |>
  step_manifold_sculpting(all_numeric_predictors(), num_comp = 2, seed = 1)

rec_trained <- prep(rec)
rec_trained
bake(rec_trained, new_data = test)
tidy(rec_trained, number = 1)
```

Index

`bake()`, 8

`manifold_sculpting`, 2
`manifold_sculpting()`, 5, 7–9

`predict.manifold_sculpting`, 5
`predict.manifold_sculpting()`, 4, 9
`prep()`, 8

`recipes::bake()`, 9
`recipes::prep()`, 8
`recipes::selections()`, 8
`recipes::step_normalize()`, 9

`set.seed()`, 4
`sim_manifolds`, 6
`sim_s_curve(sim_manifolds)`, 6
`sim_s_curve()`, 4
`sim_swiss_roll(sim_manifolds)`, 6
`sim_swiss_roll()`, 4
`squishing_rate`, 7
`squishing_rate()`, 9
`step_manifold_sculpting`, 7
`step_manifold_sculpting()`, 4, 7

`tidy()`, 9
`tidy.step_manifold_sculpting`
 (`step_manifold_sculpting`), 7