# Package ‘nnmf’

March 17, 2026

**Type** Package

**Title** Nonnegative Matrix Factorization

**Version** 1.3

**Date** 2026-03-17

**Maintainer** Michail Tsagris <mtsagris@uoc.gr>

**Depends** R (>= 4.0)

**Imports** Rcpp, ClusterR, Compositional, graphics, Matrix, osqp, parallel, quadprog, Rfast, Rfast2, Rglpk, sparcl, stats

**LinkingTo** Rcpp, RcppEigen

**Description**

Nonnegative matrix factorization (NMF) is a technique to factorize a matrix with nonnegative values into the product of two matrices. Covariates are also allowed. Parallel computing is an option to enhance the speed and high-dimensional and large scale (and/or sparse) data are allowed. Relevant papers include: Wang Y. X. and Zhang Y. J. (2012). Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 25(6): 1336-1353 <doi:10.1109/TKDE.2012.51> and Kim H. and Park H. (2008). Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 30(2): 713-730 <doi:10.1137/07069239X>.

**License** GPL (>= 2)

**Encoding** UTF-8

**NeedsCompilation** yes

**Author** Michail Tsagris [aut, cre],
Nikolaos Kontemeniotis [aut],
Christos Adam [aut] (ORCID: <https://orcid.org/0009-0003-3244-7034>)

**Repository** CRAN

**Date/Publication** 2026-03-17 10:10:02 UTC

## Contents

---

`nnmf-package` *Nonnegative Matrix Factorization*

---

## Description

Nonnegative matrix factorization (NMF) is implemented.

## Details

| | |
|---|---|
| Package: | nnmf |
| Type: | Package |
| Version: | 1.3 |
| Date: | 2026-03-17 |
| License: | GPL-2 |

## Maintainers

Michail Tsagris <mtsagris@uoc.gr>.

## Author(s)

Michail Tsagris `<mtsagris@uoc.gr>`, Nikolaos Kontemeniotis `<kontemeniotisn@gmail.com>` and Christos Adam `<pada4m4@gmail.com>`.

## References

Erichson N. B., Mendible A., Wihlborn S. and Kutz J. N. (2018). Randomized nonnegative matrix factorization. Pattern Recognition Letters, 104: 1-7.

Wang Y. X. and Zhang Y. J. (2012). Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 25(6): 1336-1353.

Kim H. and Park H. (2008). Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 30(2): 713-730.

Cutler A. and Breiman L. (1994). Archetypal analysis. Technometrics, 36(4): 338–347.

`init` *Initialization strategies for the NMF based on the k-means*

### Description

Initialization strategies for the NMF based on the k-means algorithm.

### Usage

```
init(x, k, bs = 1, veo = FALSE)
```

### Arguments

| | |
|---|---|
| `x` | An $n \times D$ numerical matrix with data. |
| `k` | The number of lower dimensions. It must be less than the dimensionality of the data, at most $D - 1$. |
| `bs` | The batch size in case the user wants to use the mini-batch k-means algorithm. If bs=1, the classical k-means is used. |
| `veo` | If the (number of) variables exceed the (number of) observations set this equal to true. In this case, the sparse k-means algorithm of Witten and Tibshirani (2010) is used to initiate the H matrix. |

### Details

Nonnegative matrix factorization using quadratic programming is performed. The objective function to be minimized is the square of the Frobenius norm.

### Value

The $H$ matrix, an $k \times D$ matrix.

### Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris `<mtsagris@uoc.gr>`.

### References

Wang Y. X. and Zhang Y. J. (2012). Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 25(6): 1336-1353.

Kim H. and Park H. (2008). Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 30(2): 713-730.

### See Also

`nmf.qp`

## Examples

```
x <- as.matrix(iris[, 1:4])
mod <- nmf.qp(x, 2)
plot(mod$W, colour = iris[, 5])
```

---

`nmf.hals` *NMF minimizing using the hierarchical ALS algorithm*

---

## Description

NMF minimizing using the hierarchical ALS algorithm.

## Usage

```
nmf.hals(x, k, maxiter = 2000, tol = 1e-6, history = FALSE)
```

## Arguments

| | |
|---|---|
| `x` | An $n \times D$ numerical matrix with data. |
| `k` | The number of lower dimensions. It must be less than the dimensionality of the data, at most $D-1$. |
| `maxiter` | The maximum number of iterations allowed. |
| `tol` | The tolerance value to terminate the quadratic programming algorithm. |
| `history` | If this is TRUE, the reconstruction error at each iteration is returned. |

## Details

Nonnegative matrix factorization using the hierarchical alternating least squares algorithm is performed. The objective function to be minimized is the square of the Frobenius norm.

## Value

| | |
|---|---|
| `W` | The $W$ matrix, an $n \times k$ matrix with the mapped data. |
| `H` | The $H$ matrix, an $k \times D$ matrix. |
| `Z` | The reconstructed data, $Z = WH$. |
| `obj` | The reconstruction error, $\|\|x - Z\|\|_F^2$. |
| `error` | If the argument history was set to TRUE the reconstruction error at each iteration will be performed, otherwise this is NULL. |
| `iters` | The number of iterations performed. |
| `runtime` | The runtime required by the algorithm. |

## Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

### References

Erichson N. B., Mendible A., Wihlborn S. and Kutz J. N. (2018). Randomized nonnegative matrix factorization. Pattern Recognition Letters, 104: 1-7. https://arxiv.org/pdf/1711.02037

### See Also

nmf.qp, nmf.sqp

### Examples

```
x <- as.matrix(iris[, 1:4])
mod <- nmf.qp(x, 2)
group <- as.numeric(iris[, 5])
plot(mod$W, col = group)
```

---

## nmf.manh — *Simplicial NMF minimizing the Manhattan distance*

---

### Description

NMF minimizing the Manhattan distance.

### Usage

```
nmf.manh(x, k, W = NULL, H = NULL, k_meds = TRUE,
maxiter = 1000, tol = 1e-6, ncores = 1)
```

### Arguments

| | |
|---|---|
| x | An $n \times D$ matrix with data. Zero values are allowed. |
| k | The number of lower dimensions. It must be less than the dimensionality of the data, at most $D - 1$. |
| W | If you have an initial estimate for W supply it here. Otherwise leave it NULL. |
| H | If you have an initial estimate for H supply it here, otherwise leave it NULL. |
| k_meds | If this is TRUE, then the K-medoids algorithm is used to initiate the W and H matrices. |
| maxiter | The maximum number of iterations allowed. |
| tol | The tolerance value to terminate the quadratic programming algorithm. |
| ncores | Do you want the update of W to be performed in parallel? If yes, specify the number of cores to use. |

### Details

Nonnegative matrix factorization minimizing the Manhattan distance.

## Value

| | |
|---|---|
| W | The $W$ matrix, an $n \times k$ matrix with the mapped data. |
| H | The $H$ matrix, an $k \times D$ matrix. |
| Z | The reconstructed data, $Z = WH$. |
| obj | The reconstruction error, $\|\|x - Z\|\|_F^2$. |
| error | If the argument history was set to TRUE the reconstruction error at each iteration will be performed, otherwise this is NULL. |
| iters | The number of iterations performed. |
| runtime | The runtime required by the algorithm. |

## Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

## References

Wang Y. X. and Zhang Y. J. (2012). Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 25(6): 1336-1353.

Kim H. and Park H. (2008). Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 30(2): 713-730.

## See Also

nmf.qp

## Examples

```
x <- as.matrix(iris[, 1:4])
mod <- nmf.qp(x, 3)
group <- as.numeric(iris[, 5])
plot(mod$W, col = group)
```

---

nmf.qp *NMF minimizing the Frobenius norm*

---

## Description

NMF minimizing the Frobenius norm using quadratic programming.

## Usage

```
nmf.qp(x, k, H = NULL, k_means = TRUE, bs = 1, veo = FALSE, lr_h = 0.1,
maxiter = 1000, tol = 1e-6, ridge = 1e-8, history = FALSE, ncores = 1)
```

## Arguments

| | |
|---|---|
| `x` | An $n \times D$ numerical matrix with data. |
| `k` | The number of lower dimensions. It must be less than the dimensionality of the data, at most $D - 1$. |
| `H` | If you have an initial estimate for H supply it here, otherwise leave it NULL. |
| `k_means` | If this is TRUE, then the K-means algorithm is used to initiate the W and H matrices. |
| `bs` | If you use the K-means algorithm for initialization, you may want to use the mini batch K-means if you have millions of observations. In this case, you need to define the number of batches. |
| `veo` | If the number of variables excceeds the number of observations set this is equal to TRUE. |
| `lr_h` | If veo is TRUE, then the exponentiated gradient descent method is used to update the H matrix. In this case you need to supply the value of the learning rate, which is 0.1 by default. |
| `maxiter` | The maximum number of iterations allowed. |
| `tol` | The tolerance value to terminate the quadratic programming algorithm. |
| `ridge` | A small quantity added in the diagonal of the $D$ matrix. |
| `history` | If this is TRUE, the reconstruction error at each iteration is returned. |
| `ncores` | Do you want the update of W to be performed in parallel? If yes, specify the number of cores to use. |

## Details

Nonnegative matrix factorization using quadratic programming is performed. The objective function to be minimized is the square of the Frobenius norm.

## Value

| | |
|---|---|
| `W` | The $W$ matrix, an $n \times k$ matrix with the mapped data. |
| `H` | The $H$ matrix, an $k \times D$ matrix. |
| `Z` | The reconstructed data, $Z = WH$. |
| `obj` | The reconstruction error, $\|\|x - Z\|\|_F^2$. |
| `error` | If the argument history was set to TRUE the reconstruction error at each iteration will be performed, otherwise this is NULL. |
| `iters` | The number of iterations performed. |
| `runtime` | The runtime required by the algorithm. |

## Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris `<mtsagris@uoc.gr>`.

### References

Wang Y. X. and Zhang Y. J. (2012). Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 25(6): 1336-1353.

Kim H. and Park H. (2008). Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 30(2): 713-730.

### See Also

nmf.hals, nmf.sqp

### Examples

```
x <- as.matrix(iris[, 1:4])
mod <- nmf.qp(x, 2)
group <- as.numeric(iris[, 5])
plot(mod$W, col = group)
```

---

## nmf.sqp *NMF minimizing the Frobenius norm*

---

### Description

NMF minimizing the Frobenius norm using sequential quadratic programming.

### Usage

```
nmf.sqp(x, k, H = NULL, maxiter = 1000, tol = 1e-4, ridge = 1e-8,
history = FALSE, ncores = 1)
```

### Arguments

| | |
|---|---|
| x | An $n \times D$ dgC class sparse matrix with data. |
| k | The number of lower dimensions. It must be less than the dimensionality of the data, at most $D - 1$. |
| H | If you have an initial estimate for H supply it here, otherwise leave it NULL. |
| maxiter | The maximum number of iterations allowed. |
| tol | The tolerance value to terminate the quadratic programming algorithm. The value is set to 1e-4 in this case because with large scale and/or sparse data the computation time is really high. So, we sacrifice some accuracy over speed. |
| ridge | A small quantity added in the diagonal of the $D$ matrix. |
| history | If this is TRUE, the reconstruction error at each iteration is returned. |
| ncores | Do you want the update of W to be performed in parallel? If yes, specify the number of cores to use. |

### Details

Nonnegative matrix factorization using quadratic programming is performed. The objective function to be minimized is the square of the Frobenius norm. This function is suitable for large scale sparse data, and parallel computing is a must in this case. Note that we do not use k-means here and that the reconstruced matrix Z is not returned with this function for capacity purposes.

### Value

| | |
|---|---|
| `W` | The $W$ matrix, an $n \times k$ matrix with the mapped data. |
| `H` | The $H$ matrix, an $k \times D$ matrix. |
| `obj` | The reconstruction error, $\|\|x - Z\|\|_F^2$. |
| `error` | If the argument history was set to TRUE the reconstruction error at each iteration will be performed, otherwise this is NULL. |
| `iters` | The number of iterations performed. |
| `runtime` | The runtime required by the algorithm. |

### Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

### References

Wang Y. X. and Zhang Y. J. (2012). Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 25(6): 1336-1353.

Kim H. and Park H. (2008). Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 30(2): 713-730.

### See Also

`nmf.qp`

### Examples

```
x <- as.matrix(iris[, 1:4])
mod <- nmf.qp(x, 2)
group <- as.numeric(iris[, 5])
plot(mod$W, col = group)
```

nmfqp.cv *K-fold cross-validation for choosing the rank in NMF*

## Description

K-fold cross-validation for choosing the rank in NMF.

## Usage

```
nmfqp.cv(x, k = 3:10, k_means = TRUE, bs = 1, veo = FALSE, lr_h = 0.1, maxiter = 1000,
tol = 1e-6, ridge = 1e-8, ncores = 1, folds = NULL, nfolds = 10, graph = FALSE)
```

## Arguments

| | |
|---|---|
| x | An $n \times D$ matrix with compositional data. Zero values are allowed. |
| k | The number of lower dimensions. It must be less than the dimensionality of the data, at most $D - 1$. |
| k_means | If this is TRUE, then the K-means algorithm is used to initiate the W and H matrices. |
| bs | If you use the K-means algorithm for initialization, you may want to use the mini batch K-means if you have millions of observations. In this case, you need to define the number of batches. |
| veo | If the number of variables excceeds the number of observations set this is equal to TRUE. In this case, the sparse k-means algorithm of Witten and Tibshirani (2010) is used to initialize the H matrix. |
| lr_h | If veo is TRUE, then the exponentiated gradient descent method is used to update the H matrix. In this case you need to supply the value of the learning rate, which is 0.1 by default. |
| maxiter | The maximum number of iterations allowed. |
| tol | The tolerance value to terminate the quadratic programming algorithm. |
| ridge | A small quantity added in the diagonal of the $D$ matrix. |
| ncores | Do you want the update of W to be performed in parallel? If yes, specify the number of cores to use. |
| folds | If you have the list with the folds supply it here. You can also leave it NULL and it will create folds. |
| nfolds | The number of folds to produce. |
| graph | If this is TRUE, the plot of the predicted error will be plotted. |

## Details

K-fold cross-validation to select the optimal rank k.

### Value

| | |
|---|---|
| sse | The matrix with the sum of squares of residuals. |
| mspe | A vector with the mean squares of residuals. |
| runtime | The runtime required by the algorithm. |

### Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

### References

Wang Y. X. and Zhang Y. J. (2012). Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 25(6): 1336-1353.

Kim H. and Park H. (2008). Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 30(2): 713-730.

### See Also

nmf.qp, nmfqp.pred

### Examples

```
x <- as.matrix(iris[1:100, 1:4])
mod <- nmfqp.cv(x, 2:3)
```

---

## nmfqp.pred *Prediction of new values using NMF*

---

### Description

Prediction of new values using NMF.

### Usage

```
nmfqp.pred(xnew, H, ridge = 1e-8, ncores = 1)
```

### Arguments

| | |
|---|---|
| xnew | An $n \times D$ numerical matrix with new data. |
| H | The H matrix produced by the NMF on the observed data. |
| ridge | A small quantity added in the diagonal of the $D$ matrix. |
| ncores | Do you want the update of W to be performed in parallel? If yes, specify the number of cores to use. |

### Details

Based on an already NMF that was produced by minimizing the square of the Frobenius norm, the function estimates the $W$ and $Z$ matrices for some new data.

### Value

| | |
|---|---|
| Wnew | The $W$ matrix for the new data, an $n \times k$ matrix with the mapped data. |
| Znew | The reconstructed new data, $Znew = WnewHnew$. |
| runtime | The runtime required by the algorithm. |

### Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

### References

Wang Y. X. and Zhang Y. J. (2012). Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 25(6): 1336-1353.

Kim H. and Park H. (2008). Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 30(2): 713-730.

### See Also

nmf.sqp

### Examples

```
x <- as.matrix(iris[1:140, 1:4])
xnew <- as.matrix(iris[141:150, 1:4])
mod <- nmf.qp(x, 2)
pred <- nmfqp.pred(xnew, mod$H)
```

---

## nmfqp.reg *NMF with covariates minimizing the Frobenius norm*

---

### Description

NMF with covariates minimizing the Frobenius norm using quadratic programming.

### Usage

```
nmfqp.reg(x, z, k, maxiter = 1000, tol = 1e-6, ncores = 1)
```

## Arguments

| | |
|---|---|
| `x` | An $n \times D$ numerical matrix with data. |
| `z` | An $n \times q$ matrix with the covariates. |
| `k` | The number of lower dimensions. It must be less than the dimensionality of the data, at most $D-1$. |
| `maxiter` | The maximum number of iterations allowed. |
| `tol` | The tolerance value to terminate the quadratic programming algorithm. |
| `ncores` | Do you want the update of W to be performed in parallel? If yes, specify the number of cores to use. |

## Details

Nonnegative matrix factorization with covariates using quadratic programming is performed. The objective function to be minimized is the square of the Frobenius norm of the residuals produced by the reconstructed matrix.

## Value

| | |
|---|---|
| `B` | The $B$ matrix, an $q \times D$ matrix with the coefficients of the covariates. |
| `W` | The $W$ matrix, an $n \times k$ matrix with the mapped data. |
| `H` | The $H$ matrix, an $k \times D$ matrix. |
| `fitted` | The reconstructed data, $fitted = ZB + WH$. |
| `obj` | The reconstruction error, $\|\|x - fitted\|\|_F^2$. |
| `iters` | The number of iterations performed. |
| `runtime` | The runtime required by the algorithm. |

## Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <`mtsagris@uoc.gr`>.

## References

Wang Y. X. and Zhang Y. J. (2012). Nonnegative matrix factorization: A comprehensive review. IEEE Transactions on Knowledge and Data Engineering, 25(6): 1336-1353.

Kim H. and Park H. (2008). Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM Journal on Matrix Analysis and Applications, 30(2): 713-730.

## See Also

`nmf.sqp`

## Examples

```
x <- as.matrix(iris[, 1:3])
z <- model.matrix(x ~., data = iris[, 4:5])[, -1]
mod <- nmfqp.reg(x, z, 2)
```

# Index