Package 'vrnmf'

October 12, 2022

Title Volume-Regularized Structured Matrix Factorization

Version 1.0.2

Description

Implements a set of routines to perform structured matrix factorization with minimum volume constraints. The NMF procedure decomposes a matrix X into a product C * D. Given conditions such that the matrix C is non-negative and has sufficiently spread columns, then volume minimization of a matrix D delivers a correct and unique, up to a scale and permutation, solution (C, D). This package provides both an implementation of volume-regularized NMF and ``anchor-free'' NMF, whereby the standard NMF problem is reformulated in the covariance domain. This algorithm was applied in Vladimir B. Seplyarskiy Ruslan A. Soldatov, et al. ``Population sequencing data reveal a compendium of mutational processes in the human germ line''. Science, 12 Aug 2021. <doi:10.1126/science.aba7408>. This package interacts with data available through the 'simulatedNMF' package, which is available in a 'drat' repository. To access this data package, see the instructions at <https://github.com/kharchenkolab/vrnmf>. The size of the 'simulatedNMF' package is approximately 8 MB.

License GPL-3

Encoding UTF-8

Depends R (>= 3.5.1)

Imports graphics, ica (>= 1.0), lpSolveAPI (>= 5.5.2.0), Matrix, nnls, parallel (>= 3.5.1), quadprog (>= 1.5), stats

Suggests knitr (>= 1.28), rmarkdown (>= 2.1), testthat

RoxygenNote 7.1.2

URL https://github.com/kharchenkolab/vrnmf

BugReports https://github.com/kharchenkolab/vrnmf/issues

Author Ruslan Soldatov [aut], Peter Kharchenko [aut], Viktor Petukhov [aut], Evan Biederstedt [cre, aut]

Maintainer Evan Biederstedt <evan.biederstedt@gmail.com>

NeedsCompilation no

Repository CRAN

Date/Publication 2022-02-25 04:20:02 UTC

R topics documented:

`AnchorFree` *Non-negative tri-factorization of co-occurence matrix using minimum volume approach.*

Description

`AnchorFree` method tri-factorizes (co-occurence) matrix in a product $P\,C{*}E{*}t(C)$ of non-negative matrices C and E such that matrix E has mininum volume and columns of matrix C equal to 1.

Usage

```
AnchorFree(
  vol,
  n.comp = 3,
  init = NULL,
  init.type = "diag",
  n.iter = 30,
  err.cut = 1e-30,
  verbose = FALSE
)
```

Arguments

`vol`	An output object of vol_preprocess(). The method factorizes co-occurence matrix `vol$P`.
`n.comp`	An integer. Number of components to extract (by default 3). Defines number of columns in matrix C. (default=3)
`init`	A numeric matrix. Initial matrix `M`. (default=3)
`init.type`	A character. A strategy to randomly initialize matrix `M`. (default="diag") Options are to 1) generate diagonal unit matrix ("diag"), 2) use ICA solution as initialization ("ica", "ica.pos"). or sample entries from: 3) uniform distribution `[0,1]` ("unif.pos"), 4) unform distribution `[-1,1]`, 5) uniform distribution `[0.9,1.1]` ("similar"), 6) normal distribution `N(0,1)`.
`n.iter`	An integer. Number of iterations. (default=30)
`err.cut`	A numeric. Relative error in determinant between iterations to stop algorithm (now is not used). (default=1e-30)
`verbose`	A boolean. Print per-iteration information (default=FALSE)

Details

Implementation closely follows (Fu X *et al.*, IEEE Trans Pattern Anal Mach Intell., 2019).

Value

List of objects:

`C`, `E` Factorization matrices.

`Pest` Estimate of `vol$P` co-occurence matrix $Pest = C * E * t(C)$.

`M`, `detM` auxiliary matrix `M` and its determinant.

`init.type` type of initialization of matrix `M` that was used.

Examples

```
small_example <- sim_factors(5, 5, 5)
vol <- vol_preprocess(t(small_example$X))
vol.anchor <- AnchorFree(vol)
```

`factor_intensities` *Infer a matrix of non-negative intensities in NMF with offset/nmf-offset.*

Description

`factor_intensities` estimates a non-negative matrix `D` that optimizes the objective function $F = ||X - C * D - offset||^2$, where offset is either column-specific offset or a "1-rank nmf term": product of row vector and column vector

Usage

```
factor_intensities(
  C,
  X,
  fit.nmf = TRUE,
  fit.factor = FALSE,
  qp.exact = FALSE,
  n.iter = 200,
  qp.iter = 10,
  rel.error.cutoff = 1e-05,
  extrapolate = TRUE,
  extrapolate.const = TRUE,
  extrapolate.convex = FALSE,
  q.factor = 1,
  verbose = TRUE,
  n.cores = 1
)
```

Arguments

`C`	Numeric matrices.
`X`	Numeric matrices.
`fit.nmf`	A boolean. Fit both intensities and spectrum of the offset residuals.
`fit.factor`	A boolean. Fit only spectrum of the offset residuals (keep intensities constant across samples).
`qp.exact`	A boolean. Estimate intensities using exact quadratic programming (qp.exact = TRUE) or inexact QP via gradient decent with extrapolation (qp.exact = FALSE).
`n.iter`	An integer. Number of iterations.
`qp.iter`	= 1e+1 An integer. Number of iterations of inexact QP.
`rel.error.cutoff`	A numeric. Relative error cutoff between iterations to stop iterations.
`extrapolate`	A boolean. Use Nesterov-like extrapolation at each iteration.

extrapolate.const
: A boolean. Use extrapolation scheme that adds a constant extrapolation q.factor (described below) at each iteration.

extrapolate.convex
: A boolean. Use Nesterov extrapolation scheme.

q.factor
: A numeric. Specification of a a constant extrapolation factor used in case of extrapolate.const = T.

verbose
: A boolean. Print per-iteration information (by default TRUE).

n.cores
: An integer. Number of cores to use.

Value

Fitted matrix D.

---

infer_intensities *Infer a matrix of non-negative intensities in NMF*

Description

infer_intensities estimates a non-negative matrix D that optimizes the objective function $F = ||X - C * D||^2$ using per-row quadratic programming.

Usage

```
infer_intensities(C, X, esign = "pos", n.cores = 1)
```

Arguments

C
: Numeric matrices.

X
: Numeric matrices.

esign
: A character. Keep elements of matrix D non-negative ("pos") or not ("all). (default="pos")

n.cores
: An integer. Number of cores to use. (default=1)

Value

Fitted matrix D.

projection_onto_simplex *Project vector onto a probabilistic simplex.*

Description

projection_onto_simplex projects a vector unproj onto a probabilistic simplex of sum bound.

Usage

```
projection_onto_simplex(unproj, bound)
```

Arguments

unproj	A numeric vector. An unprojected vector
bound	A numeric. Sum of projected vector elements.

Value

A projected vector.

sim_factors *Simulate matrices to explores* vrnmf

Description

sim_factors simulates non-negative factorization matrices C and D under a variaty of conditions to explore factorization $X = C * D + noise$.

Usage

```
sim_factors(
  m,
  n,
  r,
  simplex = "col",
  distr = "unif",
  frac.zeros = 0.4,
  condition = FALSE,
  noise = 0
)
```

Arguments

m	Integers. Size of matrices. Matrix C has a size of m*r and matrix D has a size of r*n.
n	Integers. Size of matrices. Matrix C has a size of m*r and matrix D has a size of r*n.
r	Integers. Size of matrices. Matrix C has a size of m*r and matrix D has a size of r*n.
simplex	A character. Either columns ("col") or rows ("row") of matrix C are projected onto unit simplex. (default="col")
distr	A character. Distribution to simulate matrix entries: "unif" for uniform and "exp" for exponential distributions. (default="unif")
frac.zeros	A numeric. Fraction of zeros in matrix C. It promotes sufficient scattering of matrix column/row vectors. (default=0.4)
condition	A boolean. Generate more well-conditioned matrix R. (default=FALSE)
noise	A numeric. Standard deviation of gaussian noise to add. (default=0e-4)

Value

List of simulated matrices:

X.noise, X - noisy and original matrix X to decompose.

C, D - factorization matrices.

---

volnmf_det *Update volume-regularized matrix* R *using det volume approximation*

Description

volnmf_det finds matrix R that minimizes objective ||X-C*R||^2 + w.vol*det(R)

Usage

```
volnmf_det(
  C,
  X,
  R,
  posit = FALSE,
  w.vol = 0.1,
  eigen.cut = 1e-16,
  err.cut = 0.001,
  n.iter = 1000
)
```

Arguments

C	Numeric Matrices. Matrices involved in objective function. Matrix R serves as initialization.
X	Numeric Matrices. Matrices involved in objective function. Matrix R serves as initialization.
R	Numeric Matrices. Matrices involved in objective function. Matrix R serves as initialization.
posit	A boolean. Set up (TRUE) or not (FALSE) non-negative constraints on matrix R. (default=TRUE)
w.vol	A numeric. Volume (det) weight in objective function. (default=0.1)
eigen.cut	A numeric. Threshold on eigenvalue of SVD eigenvectors. (default=1e-16)
err.cut	A numeric. Stop algorithm if relative erro in R between iteration is less than err.cut. (default=1e-3)
n.iter	An integer. Number of iterations. (default=1e+3)

Value

An updated matrix R.

---

volnmf_estimate *Alternating optimization of volume-regularized NMF*

Description

volnmf_estimate provides alternating optimization of volume-regularized factorization of a matrix B using the following objective function: $F = ||B*Q - C*R||^2 + w.vol*volume(R)$. Matrix C is required to be non-negative and having either column or row vectors on the simplex. Matrix R can optionally have non-negativity constraint. Matrix Q can optionally be identity matrix or any unitary.

Usage

```
volnmf_estimate(
  B,
  C,
  R,
  Q,
  domain = "covariance",
  volf = "logdet",
  R.majorate = FALSE,
  wvol = NULL,
  delta = 1e-08,
  n.iter = 10000,
  err.cut = 1e-08,
```

```
  vol.iter = 100,
  c.iter = 100,
  extrapolate = TRUE,
  accelerate = TRUE,
  acc.C = 4/5,
  acc.R = 3/4,
  C.constraint = "col",
  C.bound = 1,
  R.constraint = "pos",
  verbose = TRUE,
  record = 100,
  Canchor = NULL,
  Ctrue = NULL,
  mutation.run = FALSE
)
```

Arguments

B	A numeric matrix. A matrix to factorize (by default NULL). If not given than matrix B is taken to be a square root decomposition of $P = B * t(B)$.
C	Numeric matrices. Initial matrices for optimiztion.
R	Numeric matrices. Initial matrices for optimiztion.
Q	Numeric matrices. Initial matrices for optimiztion.
domain	A character. Optimize unitary rotation matrix Q ("covariance") or keep it as identity matrix (as in standard NMF). By default "covariance".
volf	A character. Function that approximate volume. Can have values of "logdet" or "det" (by default "logdet").
R.majorate	A boolean. Majorate logdet each iteration of volnmf_logdet() (by default FALSE).
wvol	A numeric. A weight of volume-regularized term volume(R).
delta	A numeric. Logdet regularization term log(det(R) + delta) (by default 1e-8).
n.iter	An integer. Number of iterations (by default 1,000).
err.cut	A numeric. Relative error in determinant between iterations to stop algorithm (by default 1e-8).
vol.iter	An integer. Number of iterations to update volume-regularized matrix R at each alternating step.
c.iter	An integer. Number of iterations to update simplex matrix C at each alternating step.
extrapolate	A numeric. Do Nesterov extrapolation inside blocks of R and C optimization (by default TRUE).
accelerate	A numeric. Do acceleration each update after R and C blocks estimated via Nesterov-like extrapolation.
acc.C	A numeric. Acceleration parameter of matrix C.
acc.R	A numeric. Acceleration parameter of matrix R.

C.constraint	A character. Constraint either sum of columns ("col") or sum of rows ("row") to be equal to C.bound (By default "col").
C.bound	A numeric. A simplex constraint on matrix C vectors.
R.constraint	A character. Set up non-negativity ("pos") constraint on elements of R (by default "pos", alternative "no").
verbose	A boolean. Print per-iteration information (by default FALSE)
record	A numeric. Record parameters every 'record' iterations (by default NULL).
Canchor	A matrix. A matrix of anchor components (unused currently). (default=NULL)
Ctrue	A matrix. Correct matrix C if known. Useful for benchmark.
mutation.run	A boolean. Assess goodness of solution using reflection test if mutation.run=TRUE (applicable only to analysis of mutation patterns). (default=FALSE)

Value

List of objects:

C, R, Q, E Factorization matrices.

iter, err Number of iterations and relative per-iteration error err in matrix C.

info.record a list of objects that record and store state of matrices each record iterations.

---

volnmf_logdet *Update volume-regularized matrix* R *using logdet volume approximation.*

Description

volnmf_logdet finds matrix R that minimizes objective ||X-C*R||^2 + w.vol*log(det(R)+delta).

Usage

```
volnmf_logdet(
  C,
  X,
  R,
  R.constraint = "pos",
  majorate = FALSE,
  extrapolate = TRUE,
  qmax = 100,
  w.vol = 0.1,
  delta = 1,
  err.cut = 0.001,
  n.iter = 1000
)
```

Arguments

C	Numeric Matrices. Matrices involved in objective function.Matrix R serves as initialization.
X	Numeric Matrices. Matrices involved in objective function.Matrix R serves as initialization.
R	Numeric Matrices. Matrices involved in objective function.Matrix R serves as initialization.
R.constraint	A character. Set up ('pos') or not ('no') non-negative constraints on matrix R (by default 'pos').
majorate	A boolean. Majorate logdet each iteration (by default FALSE).
extrapolate	A boolean. Use Nesterov acceleration (by default FALSE, currently is not supported).
qmax	A numeric. Maximum asymptotic (1 - 1/qmax) of extrapolation step.
w.vol	A numeric. Volume (logdet) weight in objective function.
delta	A numeric. Determinant pseudocount in objective function.
err.cut	A numeric. Stop algorithm if relative erro in R between iteration is less than err.cut.
n.iter	An integer. Number of iterations.

Value

An updated matrix R.

---

volnmf_main *Volume-regularized NMF*

Description

volnmf_main enables volume-regularized factorization of a matrix B using the following objective function: $F = ||B*Q - C*R||^2 + w.vol*volume(R)$. Matrix C is required to be non-negative and having either column or row vectors on the simplex. Matrix R can optionally have non-negativity constraint. Matrix Q can optionally be identity matrix or any unitary. The latter option is used to decompose co-occurence matrix vol_P.

Usage

```
volnmf_main(
  vol,
  B = NULL,
  volnmf = NULL,
  n.comp = 3,
  n.reduce = n.comp,
  do.nmf = TRUE,
```

```
  iter.nmf = 100,
  seed = NULL,
  domain = "covariance",
  volf = "logdet",
  wvol = NULL,
  delta = 1e-08,
  n.iter = 500,
  err.cut = 1e-16,
  vol.iter = 20,
  c.iter = 20,
  extrapolate = TRUE,
  accelerate = FALSE,
  acc.C = 4/5,
  acc.R = 3/4,
  C.constraint = "col",
  C.bound = 1,
  R.constraint = "pos",
  R.majorate = FALSE,
  C.init = NULL,
  R.init = NULL,
  Q.init = NULL,
  anchor = NULL,
  Ctrue = NULL,
  verbose = TRUE,
  record = 100,
  verbose.nmf = FALSE,
  record.nmf = NULL,
  mutation.run = FALSE
)
```

Arguments

vol	An output object of vol_preprocess().
B	A numeric matrix. A matrix to factorize (by default NULL). If not given than matrix B is taken to be a square root decomposition of $P = B * t(B)$.
volnmf	An output object of volnmf.main. An option is useful to re-estimate solution using different parameters (by default NULL).
n.comp	An integer. Number of components to extract (by default 3). Defines number of columns in matrix C.
n.reduce	An integer. Dimensional reduction of matrix B (number of columns) if taken as a square root decomposition of volP (by default equal to n.comp).
do.nmf	A boolean. Estimate standard solution with w.vol=0 as initialization before applying volume regularization (by default TRUE).
iter.nmf	An integer. Number of iterations to get solution with w.vol=0 if the former requested (by default 1,000).
seed	An integer. Fix seed.

domain	A character. Optimize unitary rotation matrix Q ("covariance") or keep it as identity matrix (as in standard NMF). By default "covariance".
volf	A character. Function that approximate volume. Can have values of "logdet" or "det" (by default "logdet").
wvol	A numeric. A weight of volume-regularized term volume(R).
delta	A numeric. Logdet regularization term log(det(R) + delta) (by default 1e-8).
n.iter	An integer. Number of iterations (by default 1,000).
err.cut	A numeric. Relative error in determinant between iterations to stop algorithm (by default 1e-8).
vol.iter	An integer. Number of iterations to update volume-regularized matrix R at each alternating step.
c.iter	An integer. Number of iterations to update simplex matrix C at each alternating step.
extrapolate	A numeric. Do Nesterov extrapolation inside blocks of R and C optimization (by default TRUE).
accelerate	A numeric. Do acceleration each update after R and C blocks estimated via Nesterov-like extrapolation.
acc.C	A numeric. Acceleration parameter of matrix C.
acc.R	A numeric. Acceleration parameter of matrix R.
C.constraint	A character. Constraint either sum of columns ("col") or sum of rows ("row) to be equal to C.bound (By default "col").
C.bound	A numeric. A simplex constraint on matrix C vectors.
R.constraint	A character. Set up non-negativity ("pos") constraint on elements of R (by default "pos", alternative "no").
R.majorate	A boolean. Majorate logdet each iteration of volnmf_logdet() (by default FALSE).
C.init	Numeric matrices. Initialization of matrices C, R, Q (by default NULL).
R.init	Numeric matrices. Initialization of matrices C, R, Q (by default NULL).
Q.init	Numeric matrices. Initialization of matrices C, R, Q (by default NULL).
anchor	An output object of AnchorFree(). Object is used optionally to initialize matrices (by default NULL).
Ctrue	A matrix. Correct matrix C if known. Useful for benchmark.
verbose	A boolean. Print per-iteration information (by default FALSE).
record	A numeric. Record parameters every 'record' iterations (by default NULL).
verbose.nmf	A boolean. Print per-iteration information for standard NMF (by default FALSE).
record.nmf	A numeric. Record parameters every 'record' iterations for standard NMF (by default NULL).
mutation.run	A boolean. Assess goodness of solution using reflection test if mutation.run=TRUE (applicable only to analysis of mutation patterns).

Value

List of objects:

`C`, `R`, `Q` Factorization matrices.

`C.init`, `R.init`, `Q.init` Initialization matrices for volume-regularized optimization.

`C.rand`, `R.rand`, `Q.rand` Random initialization matrices for NMF optimization (`w.vol=0`).

`rec` a list of objects that record and store state of matrices each `record` iterations.

`volnmf_procrustes` *Procrustes algorithm estimates orthonormal transformation between two matrices.*

Description

`volnmf_procrustes` finds orthonormal matrix `Q` that minimizes objective `||A-B*Q||^2`

Usage

```
volnmf_procrustes(A, B)
```

Arguments

`A`	Numeric Matrices. Orthonormal transformation convert matrix `B` in matrix `A`.
`B`	Numeric Matrices. Orthonormal transformation convert matrix `B` in matrix `A`.

Value

An optimal orthonormal tranformation matrix `Q`.

`volnmf_simplex_col` *Update of a matrix in NMF with equality contstraints on columns.*

Description

`volnmf_simplex_col` finds non-negative matrix `C` that minimizes the objective `||X-C*R||^2` under constraints that columns of C equal to 1 using local approximation with extrapolation.

Usage

```
volnmf_simplex_col(
  X,
  R,
  C.prev = NULL,
  bound = 1,
  extrapolate = TRUE,
  err.cut = 1e-10,
  n.iter = 10000,
  qmax = 100
)
```

Arguments

X	Numeric Matrices. Matrices involved in the objective function.
R	Numeric Matrices. Matrices involved in the objective function.
C.prev	Numeric Matrices. Matrices involved in the objective function. Matrix C.prev serves as initialization. (default=NULL)
bound	A numeric. Equality constraint on columns of matrix C. (default=1)
extrapolate	A boolean. Use extrapolation after local approximation. (default=TRUE)
err.cut	A numeric. Stop iterations if relative error between iterations is less than err.cut (parameter is not active now). (default=1e-10)
n.iter	An integer. Number of iterations. (default=1000)
qmax	A numeric. Maximum asymptotic (1 - 1/qmax) of extrapolation step.

Value

An updated matrix C.

---

volnmf_simplex_row *Update of a matrix in NMF with equality contstraints on rows.*

Description

volnmf_simplex_row finds non-negative matrix C that minimizes the objective ||X-C*R||^2 under constraints that rows of C equal to 1 using per-row quadratic programming.

Usage

```
volnmf_simplex_row(X, R, C.prev = NULL, meq = 1)
```

Arguments

X	Numeric Matrices. Matrices involved in the objective function.
R	Numeric Matrices. Matrices involved in the objective function.
C.prev	Numeric Matrices. Matrices involved in the objective function. Matrix C.prev serves as initialization. (default=NULL)
meq	An integer 0 or 1. Require equality (meq=1) or inequality (meq=0) constratint on rows (by default 1).

Value

An updated matrix C.

---

vol_preprocess *Preprocess the data for downstream volume analysis.*

Description

vol_preprocess Routine normalizes the data (as requested), estimates covariance and SVD decomposition.

Usage

```
vol_preprocess(X, col.norm = "sd", row.norm = NULL, pfactor = NULL)
```

Arguments

X	A numeric matrix. Covariance is estimated for column vectors of X.
col.norm	A character. Specifies column normalization strategy (by default "sd"). NULL to avoid normalization.
row.norm	A character. Specifies row normalization strategy (by default NULL).
pfactor	A numeric A factor to normalize co-occurence matrix (by default NULL). Row normalization follows column normalization. NULL to avoid normalization.

Value

A list of objects that include normalized matrix X.process, row and column normalization factors row.factors and col.factors, covariance matrix P0, covariance matrix P normalized to maximum value pfactor, orthonormal basis U and vector of eigenvalues eigens.

Examples

```
small_example <- sim_factors(5, 5, 5)
vol <- vol_preprocess(t(small_example$X))
```

Index