

# Package ‘tensorMQR1’

January 6, 2020

**Type** Package

**Title** tensorMQR1

**Version** 0.1.0

**Date** 2019-09-01

**Author** Xu Liu [aut,cre],  
Jian Huang [aut],  
Heng Lian [aut],  
Xinyuan Song [aut],  
Yuling Jiao [aut]

**Maintainer** Xu Liu <liu.xu@sufe.edu.cn>

**Description** Symmetric tensor estimation for multiresponse quadratic regression. The number of predictors can be diverged as sample size increases, in which the penalty LASSO, MCP or SCAD can be used.

**License** GPL (>= 2)

**Imports** Rcpp (>= 0.11.15), RcppEigen (>= 0.3.2.3.0)

**LinkingTo** Rcpp, RcppEigen

**RoxygenNote** 6.0.1

**NeedsCompilation** yes

**Repository** github

**URL** <https://github.com/xliusufe/tensorMQR1>

**Encoding** UTF-8

## R topics documented:

tensorMQR1-package . . . . .	2
generateData . . . . .	2
mqr . . . . .	3
mqr_dr . . . . .	5
mqr_sparse . . . . .	6
mqr_sparse_dr . . . . .	8
TransferModalUnfoldings . . . . .	11
TransferP2T . . . . .	12
TransferT2P . . . . .	13
<b>Index</b>	<b>14</b>

---

tensorMQR1-package	<i>Symmetric Tensor Estimation for Quadratic Regression.</i>
--------------------	--

---

### Description

For a high-dimensional multiresponse quadratic regression (MQR) with or without aparsity assumptions, treating the coefficients as a third-order tensor and borrowing Tucker decomposition to reduce the number of parameters. The multivariate sparse group lasso (mcp or scad) and the steepest gradient descent algorithm are used to estimate tensor for sparsity situation.

### Details

This section should provide a more detailed overview of how to use the package, including the most important functions.

### Author(s)

Xu Liu

Maintainer: Xu Liu <liu.xu@sufe.edu.cn>

### References

Symmetric Tensor Estimation for Quadratic Regression.

---

generateData	<i>Generate data from MQR model.</i>
--------------	--------------------------------------

---

### Description

Generate data for a high-dimensional multiresponse quadratic regression, with or without aparsity assumptions.

### Usage

```
generateData(n, q, s, p, D3, SigmaX=diag(p-1), sigma2=0.2, seed_id, t=0.0, rho=0.0)
```

### Arguments

n	Sample size.
q	The number of responses, $q \geq 1$ .
s	The true covariates associating to response, $s \geq 1$ .
p	The number of covariates, $p \geq 1$ .
D3	The mode of unfolding $D_{(3)}$ .
SigmaX	Covariance of $X$ . Default is identity matrix.
sigma2	err variance. Default is 0.1.
seed_id	Seed of generator.
rho	The correlation of $\epsilon_j$ and $\epsilon_k$ , where $j, k \in \{1, \dots, q\}$ .

## Details

This function gives  $qp(p+1)/2$  coefficients' estimators of MQR. The core tensor is a  $r_1 \times r_2 \times r_3$ -tensor. We choose  $r_1, r_2$  and  $r_3$  by BIC or CV.

## Value

Y                      Response, a  $n \times q$ -matrix.  
 X                      Design matrix, a  $n \times p$ -matrix.

## References

Symmetric Tensor Estimation for Quadratic Regression.

## See Also

mam\_sparse

## Examples

```
# Example 1

D3 <- matrix(runif(72, 0.7, 1), 2, 36)
mydata <- generateData(200, 2, 6, 6, D3)

Y <- mydata$Y
X <- mydata$X


# Example 2
n <- 500
p <- 10
q <- 10
s <- 7
s0 <- s
r10=r20=r30=2
S3 <- matrix(runif(r10*r20*r30,3,7),nrow = r30)
T1 <- matrix(rnorm(s0*r10),nrow = s0)
U1 <- qr.Q(qr(T1))
T1 <- matrix(rnorm(q*r30),nrow = q)
U3 <- qr.Q(qr(T1))
D3 <- U3%*%S3%*%t(kronecker(U1,U1))
mydata <- generateData(n,q,s0,p,D3)
```

---

mqr

---

*Fit MQR without sparsity assumption and with fixed ranks.*


---

## Description

Fit a low-dimensional multiresponse quadratic regression without aparsity assumptions and with given  $r_1, r_2, r_3$ . The steepest gradient descent algorithm is used to estimate tensor.

## Usage

```
mqr(Y, X, r1 = NULL, r3 = NULL, SUV = NULL, isSym = TRUE,
    eps = 1e-6, max_step = 20, max_step1=20)
```

## Arguments

Y	A $n \times q$ numeric matrix of responses.
X	A $n \times p$ numeric design matrix for the model.
r1	The first dimension of single value matrix of the tensor. Default is 2.
r3	The third dimension of single value matrix of the tensor. Default is 2.
SUV	A user-specified list of initial coefficient matrix of $S$ , $U$ , $V$ . By default, initial matrices are provided randomly.
isSym	A logical value indicating whether restrict tensor to be symmetric. If isSym is TRUE (the default), the core tensor is symmetric, and both $U$ and $V$ belong to Stiefel manifold. If isSym is FALSE, we decompose the tensor to $S$ , $A$ , $B$ , $C$ , where the tensor is treated to be asymmetric.
eps	Convergence threshold. The algorithm iterates until the loss function change in any coefficient is less than eps. Default is $1e-6$ .
maxstep	The maximum iterates number of the steepest gradient descent method. Default is 20.
max_step1	Maximum number of outer iterations. Default is 20.

## Details

This function gives  $qp(p+1)/2$  coefficients' estimators of MQR. The core tensor is a  $r_1 \times r_2 \times r_3$ -tensor. We fixed  $r_1$ ,  $r_2$  and  $r_3$  in the function mqr, but one can choose  $r_1$ ,  $r_2$  and  $r_3$  by BIC or CV. See details in function mqr\_bic or mqr\_cv.

## Value

Dnew	Estimator of $D_{(3)}$ .
rss	Residual sum of squares (RSS).
Y	Response $Y$ .
X	Design matrix $X$ .

## References

Symmetric Tensor Estimation for Quadratic Regression.

## See Also

mqr\_sparse, mqr\_bic, mqr\_cv

## Examples

```
D3 <- matrix(runif(72, 0.7, 1), 2, 36) # tensor with size 6*6*2
mydata <- generateData(200, 2, 6, 6, D3)

fit <- mqr(mydata$Y, mydata$X, r1=4, r3= 2)
D3hat <- fit$Dnew
```

---

mqr_dr	<i>Fit MQR without sparsity assumption, and with ranks selected by BIC or CV.</i>
--------	---

---

## Description

Fit a low-dimensional multiresponse quadratic regression without sparsity assumptions and with ranks  $r_1, r_3$  selected by BIC (the default), AIC, EBIC, CV, or GCV..

## Usage

```
mqr_dr(Y, X, r1_index = NULL, r3_index = NULL, method = "BIC", ncv = 10,
        SUV = NULL, isSym = TRUE, eps = 1e-6, max_step = 20, max_step1 = 20)
```

## Arguments

Y	A $n \times q$ numeric matrix of responses.
X	A $n \times p$ numeric design matrix for the model.
r1_index	A user-specified sequence of $r_1$ values, where $r_1$ is the first dimension of single value matrix of the tensor. Default is $r1\_index = 1, \dots, \min(\log(n), p)$ .
r3_index	A user-specified sequence of $r_3$ values, where $r_3$ is the third dimension of single value matrix of the tensor. Default is $r3\_index = 1, \dots, \min(\log(n), q)$ .
method	The method to be applied to select parameters. Either BIC (the default), AIC, EBIC, CV, or GCV.
ncv	The number of cross-validation folds. Default is 10. If method is not "CV", ncv is useless.
SUV	A user-specified list of initial coefficient matrix of $S, U, V$ , which is a list with values $S, U, V$ . By default, initial matrices are provided randomly.
isSym	A logical value indicating whether restrict tensor to be symmetric. If isSym is TRUE (the default), the core tensor is symmetric, and both $U$ and $V$ belong to Stiefel manifold. If isSym is FALSE, we decompose the tensor to $S, A, B, C$ , where the tensor is treated to be asymmetric.
eps	Convergence threshold. The algorithm iterates until the relative change in any coefficient is less than eps. Default is $1e-6$ .
maxstep	The maximum iterates number of the steepest gradient descent method. Default is 20.
max_step1	Maximum number of outer iterations. Default is 20.

## Details

This function gives  $qp(p+1)/2$  coefficients' estimators of MQR. The core tensor is a  $r_1 \times r_2 \times r_3$ -tensor. We choose  $r_1, r_2$  and  $r_3$  by BIC or CV.

**Value**

Dnew	Estimator of $D_{(3)}$ .
rss	Residual sum of squares (RSS).
rk_opt	The optimal parametres that slected by BIC or CV. It is a vector with length 4, which are selected $r_1$ and $r_3$ .
selected	Which $\lambda$ is selection.
Y	Response $Y$ .
X	Design matrix $X$ .

**References**

Symmetric Tensor Estimation for Quadratic Regression.

**See Also**

mqr, mqr\_sparse\_dr

**Examples**

```
D3 <- matrix(runif(72, 0.7, 1), 2, 36) # tensor with size 6*6*2
mydata <- generateData(200, 2, 6, 6, D3)

fit <- mqr_dr(mydata$Y, mydata$X)
D3hat <- fit$Dnew
opt <- fit$rk_opt
```

---

mqr\_sparse

---

*Fit MQR with sparsity assumption and fixed ranks.*


---

**Description**

Fit a high-dimensional multiresponse quadratic regression with or without aparsity assumptions, and given ranks given ranks  $r_1, r_2, r_3$ . The multivariate sparse group lasso (mcp or scad) and the steepest gradient descent algorithm are used to estimate tensor for sparsity situation. The tuning parameter is selected by BIC (the default), AIC, EBIC, CV, or GCV.

**Usage**

```
mqr_sparse(Y, X, r1 = NULL, r3 = NULL, method = "BIC", ncv = 10, isPenU = 0,
  isPenColumn = 1, penalty = "LASSO", lambda = NULL, SUV = NULL,
  nlam = 50, lam_min = 1e-4, ftol = 1e-6, maxstep = 20, maxstep1 = 20,
  eps = 1e-4, thresh=1e-4, gamma_pen = 2, dfmax = NULL, alpha = 1)
```

**Arguments**

<code>Y</code>	A $n \times q$ numeric matrix of responses.
<code>X</code>	A $n \times q$ numeric design matrix for the model.
<code>r1</code>	The first dimension of single value matrix of the tensor. Default is 2.
<code>r3</code>	The third dimension of single value matrix of the tensor. Default is 2.
<code>method</code>	The method to be applied to select parameters. Either BIC (the default), AIC, EBIC, CV, or GCV.
<code>ncv</code>	The number of cross-validation folds. Default is 10. If method is not CV, ncv is useless.
<code>isPenU</code>	A logical value indicating whether the rows of $U$ is penalized. Default is FALSE. If <code>isPenU</code> is FALSE, the coefficients associating with $X_j$ is penalized for each $j \in \{1, \dots, p\}$ .
<code>isPenColumn</code>	A logical value indicating whether the coefficients associating with $X_j$ that affects whole response $y$ is penalized. Default is TRUE. If <code>isPenU</code> is TRUE, the coefficients associating with $X_j$ that affects whole response $y$ is penalized for each $j \in \{1, \dots, p\}$ . If <code>isPenU</code> is FALSE, the coefficients associating with $X_j$ that affects single response $y_l$ is penalized for each $j \in \{1, \dots, p\}$ , where $l \in \{1, \dots, q\}$ .
<code>penalty</code>	The penalty to be applied to the model. Either "LASSO" (the default), "SCAD", or "MCP".
<code>lambda</code>	A user-specified sequence of lambda values. By default, a sequence of values of length <code>nlam</code> is computed, equally spaced on the log scale.
<code>SUV</code>	A user-specified list of initial coefficient matrix of $S$ , $U$ , $V$ . By default, initial matrices are provided randomly.
<code>nlam</code>	The number of lambda values. Default is 20.
<code>lam_min</code>	The smallest value for lambda, as a fraction of <code>lambda.max</code> . Default is 1e-3.
<code>ftol</code>	Convergence threshold for the Curvilinear search. The algorithm iterates until the relative change in any coefficient is less than <code>eps1</code> . Default is 1e-6.
<code>maxstep</code>	The maximum iterates number of the steepest gradient descent method. Default is 20.
<code>max_step1</code>	Maximum number of outer iterations. Default is 20.
<code>thresh</code>	The threshold to numerically determine which coefficients are zeros. Since the steepest projected gradient descent method with the approximated penalty can not shrink the estimated row of true zero row of $U$ to exactly zero, we need to determine a numerical threshold. Default is 1e-6.
<code>eps</code>	Convergence threshold for the outer loop. The algorithm iterates until the relative change in any coefficient is less than <code>eps1</code> . Default is 1e-4.
<code>gamma_pen</code>	The tuning parameter of the MCP/SCAD penalty (see details).
<code>dfmax</code>	Upper bound for the number of nonzero coefficients. Default is no upper bound. However, for large data sets, computational burden may be heavy for models with a large number of nonzero coefficients.
<code>alpha</code>	Tuning parameter for the Mnet estimator which controls the relative contributions from the LASSO, MCP/SCAD penalty and the ridge, or L2 penalty. <code>alpha=1</code> is equivalent to LASSO, MCP/SCAD penalty, while <code>alpha=0</code> would be equivalent to ridge regression. However, <code>alpha=0</code> is not supported; <code>alpha</code> may be arbitrarily small, but not exactly 0.

## Details

This function gives  $qp(p+1)/2$  coefficients' estimators of MQR. The core tensor is a  $r_1 \times r_1 \times r_3$ -tensor.  $r_1$  and  $r_3$  are fixed.

## Value

betapath	Solution path of $\beta$ .
rss	Residual sum of squares (RSS).
df	Degrees of freedom.
lambda	The sequence of regularization parameter values in the path.
lambda_opt	The value of lambda with the minimum BIC value.
selectedID	The index of lambda corresponding to lambda_opt.
activeF	The active set of $U$ . If isPenColumn is TRUE, activeF is same as activeX
activeX	The active set of coefficients associating with $X$ . If isPenColumn is TRUE, activeX is same as activeF
Snew	Estimator of $S_3$ .
Unew	Estimator of $U$ .
Vnew	Estimator of $V$ .
Y	Response $Y$ .
X	Design matrix $X$ .

## References

Symmetric Tensor Estimation for Quadratic Regression.

## See Also

mqr, mqr\_sparse\_dr

## Examples

```
D3 <- matrix(runif(72, 0.7, 1), 2, 36) # tensor with size 6*6*2
mydata <- generateData(200, 2, 6, 6, D3)

fit <- mqr_sparse(mydata$Y, mydata$X)
activeX <- fit$activeX
```

---

mqr\_sparse\_dr

---

*Fit MQR with sparsity assumption and ranks selected by BIC or CV.*


---

## Description

Fit a high-dimensional multiresponse quadratic regression with or with aparsity assumptions. The multivariate sparse group lasso (mcp or scad) and the steepest gradient descent algorithm are used to estimate tensor for sparsity situation. The tuning parameter and ranks are selected by BIC (the default), AIC, EBIC, CV, or GCV, which matchs the method of rank selection.



**Usage**

```
mqr_sparse_dr(Y, X, r1_index = NULL, r3_index = NULL, method = "BIC", ncv = 10,
  penalty = "LASSO", isPenU = 0, isPenColumn = 1, lambda = NULL, SUV = NULL,
  nlam = 50, lam_min = 0.001, ftol = 1e-6, maxstep = 20, maxstep1 = 20,
  eps = 1e-4, thresh = 1e-4, gamma_pen = 2, dfmax = NULL, alpha = 1)
```

**Arguments**

<code>Y</code>	A $n \times q$ numeric matrix of responses.
<code>X</code>	A $n \times q$ numeric design matrix for the model.
<code>r1_index</code>	A user-specified sequence of $r_1$ values, where $r_1$ is the first dimension of single value matrix of the tensor. Default is <code>r1_index = 1, ..., min(⌈log(n)⌉, p)</code> .
<code>r3_index</code>	A user-specified sequence of $r_3$ values, where $r_3$ is the third dimension of single value matrix of the tensor. Default is <code>r3_index = 1, ..., min(⌈log(n)⌉, q)</code> .
<code>method</code>	The method to be applied to select parameters. Either BIC (the default), AIC, EBIC, CV, or GCV.
<code>ncv</code>	The number of cross-validation folds. Default is 10. If method is not CV, ncv is useless.
<code>penalty</code>	The penalty to be applied to the model. Either "LASSO" (the default), "SCAD", or "MCP".
<code>isPenU</code>	A logical value indicating whether the rows of $U$ is penalized. Default is FALSE. If <code>isPenU</code> is FALSE, the coefficients associating with $X_j$ is penalized for each $j \in \{1, \dots, p\}$ .
<code>isPenColumn</code>	A logical value indicating whether the coefficients associating with $X_j$ that affects whole response $y$ is penalized. Default is TRUE. If <code>isPenU</code> is TRUE, the coefficients associating with $X_j$ that affects whole response $y$ is penalized for each $j \in \{1, \dots, p\}$ . If <code>isPenU</code> is FALSE, the coefficients associating with $X_j$ that affects single response $y_l$ is penalized for each $j \in \{1, \dots, p\}$ , where $l \in \{1, \dots, q\}$ .
<code>lambda</code>	A user-specified sequence of lambda values. By default, a sequence of values of length <code>nlam</code> is computed, equally spaced on the log scale.
<code>SUV</code>	A user-specified list of initial coefficient matrix of $S$ , $U$ , $V$ . By default, initial matrices are provided randomly.
<code>nlam</code>	The number of lambda values. Default is 50.
<code>lam_min</code>	The smallest value for lambda, as a fraction of <code>lambda.max</code> . Default is 1e-2.
<code>ftol</code>	Convergence threshold for the Curvilinear search. The algorithm iterates until the relative change in any coefficient is less than <code>eps1</code> . Default is 1e-6.
<code>maxstep</code>	The maximum iterates number of the steepest gradient descent method. Default is 20.
<code>max_step1</code>	Maximum number of outer iterations. Default is 20.
<code>eps</code>	Convergence threshold for the outer loop. The algorithm iterates until the relative change in any coefficient is less than <code>eps1</code> . Default is 1e-4.
<code>thresh</code>	The threshold to numerically determine which coefficients are zeros. Since the steepest projected gradient descent method with the approximated penalty can not shrink the estimated row of true zero row of $U$ to exactly zero, we need to determine a numerical threshold. Default is 1e-6.
<code>gamma_pen</code>	The tuning parameter of the MCP/SCAD penalty (see details).

dfmax	Upper bound for the number of nonzero coefficients. Default is no upper bound. However, for large data sets, computational burden may be heavy for models with a large number of nonzero coefficients.
alpha	Tuning parameter for the Mnet estimator which controls the relative contributions from the LASSO, MCP/SCAD penalty and the ridge, or L2 penalty. $\alpha=1$ is equivalent to LASSO, MCP/SCAD penalty, while $\alpha=0$ would be equivalent to ridge regression. However, $\alpha=0$ is not supported; $\alpha$ may be arbitrarily small, but not exactly 0.

## Details

This function gives  $qp(p+1)/2$  coefficients' estimators of MAM. The core tensor is a  $r_1 \times r_1 \times r_3$ -tensor. We choose  $r_1$  and  $r_3$  by BIC or CV.

## Value

rss	Residual sum of squares (RSS).
df	Degrees of freedom.
activeF	The active set of $U$ . If isPenColumn is TRUE, activeF is same as activeX
activeX	The active set of coefficients associating with $X$ . If isPenColumn is TRUE, activeX is same as activeF
Snew	Estimator of $S_3$ .
Unew	Estimator of $U$ .
Vnew	Estimator of $V$ .
lambda	The sequence of regularization parameter values in the path.
selectedID	The index of lambda corresponding to lambda_opt.
lambda_opt	The value of lambda with the minimum BIC or CV value.
RSS	The values of BIC or CV, which is a vector.
rk_opt	The optimal parameters that selected by BIC or CV. It is a vector with length 2, which are selected $r_1$ and $r_3$ .
Y	Response $Y$ .
X	Design matrix $X$ .

## References

Symmetric Tensor Estimation for Quadratic Regression.

## See Also

mqr\_dr, mqr\_sparse

## Examples

```
#Example 1

D3 <- matrix(runif(72, 0.7, 1), 2, 36) # tensor with size 6*6*2
mydata <- generateData(200, 2, 6, 6, D3)

fit <- mqr_sparse_dr(mydata$Y, mydata$X)
S3hat <- fit$Snew
opt <- fit$rk_opt
```

---

TransferModalUnfoldings

*Transfer a tensor's modal unfoldings to another.*


---

## Description

Transfer a tensor's modal unfoldings to another.

## Usage

```
TransferModalUnfoldings(S, d1, d2 , r1, r2, r3)
```

## Arguments

S	A mode-d1-unfolding of a tensor with size $r_1 \times r_2 \times r_3$ , input unfolding
d1	An integer, the mode of unfolding $S_{(d_1)}$
d2	An integer, the mode of output unfolding $S_{(d_2)}$
r1	The first dimension of tensor
r2	The second dimension of tensor
r3	The third dimension of tensor

## Details

This function transfers an input mode-d1-unfolding  $S_{(d_1)}$  to mode-d2-unfolding  $S_{(d_2)}$

## Value

D The output mode-d2-unfolding,  $S_{(d_2)}$

## References

Symmetric Tensor Estimation for Quadratic Regression.

## Examples

```
D1 <- matrix(1:24,nrow = 4) # A tensor unfolding with size 4*6
D2 <- TransferModalUnfoldings(D1,1,2,4,3,2)
```

---

TransferP2T	<i>Transfer coefficients of the multivariate quadratic model to a tensor's modal unfoldings.</i>
-------------	--

---

## Description

Transfer coefficients of the multivariate quadratic model to a tensor's modal unfoldings.

## Usage

```
TransferP2T(coef, d , p, q)
```

## Arguments

coef	The coefficients of the multivariate quadratic model, which is a vector with length $qp(p+1)/2$
d	An integer, the mode of unfolding $S_{(d)}$
p	The first dimension of tensor
q	The third dimension of tensor

## Details

This function transfers coefficients of the multivariate quadratic model `coef` to a mode- $d$ -unfolding  $D_{(d)}$  of a tensor.

## Value

Dd	A mode- $d$ -unfolding of a tensor with size $p \times pq$ , input unfolding.
----	---

## References

Symmetric Tensor Estimation for Quadratic Regression.

## Examples

```
p <- 4
q <- 3
D1 <- NULL
for(j in 1:q){
  D0 <- matrix(runif(p^2,1,3),p)
  D1 <- cbind(D1,(D0+t(D0))/2)
}
coef <- TransferT2P(D1, 1 , p, q)
D1 <- TransferP2T(coef, 1 , p, q)
D2 <- TransferP2T(coef, 2 , p, q)
coef2 <- TransferT2P(D2, 2 , p, q)
```

---

TransferT2P	<i>Transfer a tensor's modal unfoldings to coefficients of the multivariate quadratic model.</i>
-------------	--

---

## Description

Transfer a tensor's modal unfoldings to coefficients of the multivariate quadratic model.

## Usage

```
TransferT2P(S, d , p, q)
```

## Arguments

S	A mode- $d$ -unfolding of a tensor with size $p \times pq$ , input unfolding
d	An integer, the mode of unfolding $S_{(d)}$
p	The first dimension of tensor
q	The third dimension of tensor

## Details

This function transfers an input mode- $d$ -unfolding  $S_{(d)}$  to coefficients of the multivariate quadratic model `coef`.

## Value

<code>coef</code>	The coefficients of the multivariate quadratic model. <code>coef</code> is a vector with length $qp(p+1)/2$ .
-------------------	---

## References

Symmetric Tensor Estimation for Quadratic Regression.

## Examples

```
p <- 4
q <- 3
D1 <- NULL
for(j in 1:q){
  D0 <- matrix(runif(p^2,1,3),p)
  D1 <- cbind(D1,(D0+t(D0))/2)
}
coef <- TransferT2P(D1, 1 , p, q)
```

# Index

\*Topic **High-dimensional; Gradient descent; Stiefel manifold; Tensor estimation; Tucker decomposition.**

generateData, [2](#)  
mqr, [3](#)  
mqr\_dr, [5](#)  
mqr\_sparse, [6](#)  
mqr\_sparse\_dr, [8](#)  
tensorMQR1-package, [2](#)  
TransferModalUnfoldings, [11](#)  
TransferP2T, [12](#)  
TransferT2P, [13](#)

generateData, [2](#)

mqr, [3](#)  
mqr-function (mqr), [3](#)  
mqr\_dr, [5](#)  
mqr\_dr-function (mqr\_dr), [5](#)  
mqr\_sparse, [6](#)  
mqr\_sparse-function (mqr\_sparse), [6](#)  
mqr\_sparse\_dr, [8](#)  
mqr\_sparse\_dr-function (mqr\_sparse\_dr),  
[8](#)

tensorMQR1 (tensorMQR1-package), [2](#)  
tensorMQR1-package, [2](#)  
TransferModalUnfoldings, [11](#)  
TransferModalUnfoldings-function  
(TransferModalUnfoldings), [11](#)  
TransferP2T, [12](#)  
TransferP2T-function (TransferP2T), [12](#)  
TransferT2P, [13](#)  
TransferT2P-function (TransferT2P), [13](#)