Package ‘cartograflow’

October 12, 2022

Title Filtering Matrix for Flow Mapping
Version 1.0.3
Description Functions to prepare and filter an origin-destination matrix for thematic flow mapping purposes. This comes after Bahoken, Francoise (2016), Mapping flow matrix a contribution, PhD in Geography - Territorial sciences. See Bahoken (2017) <doi:10.4000/netcom.2565>.

Depends R (>= 3.4.0)
License GPL-3
URL https://github.com/fbahoken/cartogRaflow
Encoding UTF-8
LazyData true
RoxygenNote 7.1.0
Suggests knitr, rmarkdown, testthat, cartography
VignetteBuilder knitr,
Imports dplyr, graphics, ggplot2, maptools, plotly, rgeos, reshape2,
       rlang, sp, sf, utils, igraph
NeedsCompilation no
Author Sylvain Blondeau [aut, cre],
       Francoise Bahoken [aut]
Maintainer Sylvain Blondeau <blondeau.sylvain@yahoo.fr>
Repository CRAN
Date/Publication 2020-07-19 19:50:02 UTC

R topics documented:
cartograflow ......................................................... 2
flowanalysis ......................................................... 2
flowcarre ............................................................ 4
flowcontig .......................................................... 5
flowdist ............................................................ 6
**Description**

This package contains functions to prepare and filter origin-destination matrix for thematic flow mapping purposes. The spatial objects processing are those of sf. This comes after Bahoken, Francoise (2016) Contribution à la cartographie d’une matrice de flux, Phd in Geography, Sorbonne Paris Cité, Paris 7.

**Details**

To learn more about cartograflow, see the vignette cartograflow.html

Main functions :

- `flowanalysis` `flowcarre` `flowcontig` `flowdist` `flowgini` `flowjointure` `flowmap` `flowreduct` `flowstructmat` `flowcontig`

**flowanalysis**

Computation of a global concentration criterion of flows values or features

**Description**

Computation of a global selection criterion for filtering flows values or flow features. To be use after `flowgini` and before `flowmap`.

**Usage**

`flowanalysis(tab, fij = NULL, critflow, critlink, result)`
flowanalysis

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tab</td>
<td>input flow dataset from flowgini</td>
</tr>
<tr>
<td>fij</td>
<td>flow value between origin and destination places</td>
</tr>
<tr>
<td>critflow</td>
<td>desired level of information significativity. See Details.</td>
</tr>
<tr>
<td>critlink</td>
<td>desired level of features density. See Details.</td>
</tr>
<tr>
<td>result</td>
<td>resulting filtering criterion value. See Details.</td>
</tr>
</tbody>
</table>

Details

-critflow = desired level of flow’s information significativity (e.g. 80)
-critlink = desired level of flow’s features density (e.g. 20)
of the flow features that represents the more significant information.

-result="density" returns the desired level of features density as a
-result = "significativity" returns the level of flow significativity as a

References


Examples

library(cartograflow)
data(flowdata)

# 1/4: Computes Gini's coefficient
tabgini <- flowgini(ODpts = flows, origin = "i", destination = "j",
                   valflow = "Fij", lorenz.plot = FALSE)
### [1] Gini's coefficient = 73.16 %

# 2/4: Plot Lorenz curve
flowgini(tab_gini,
         format = "L", origin = "i", dest = "j", valflow = "ydata",
         bkg, code = "EPT_NUM", lorenz.plot = TRUE
)

# 3/4: Compute critflow filtering parameter
# critflow = 0.8 #selected criterion
flowanalysis(tabgini, critflow = 0.8, result = "signif")
### [1] "threshold = 11238 --- flows = 80 % --- links = 22.94 %"

# 4/4: Plot the flowmap
flowmap(
    tab = tabflow, fij = "Fij", origin.f = "i", destination.f = "j",
    bkg = map, code = "EPT_NUM", nodes.X = "X", nodes.Y = "Y",
    filter = TRUE,
    threshold = 11238,
flowcarre

Builds a square matrice from geographical nodes

Description

Builds a square and closed matrice from a dataframe of spatial nodes

Usage

flowcarre(liste, tab, origin, dest, valflow, empty.sq, format, diagonale)

Arguments

- liste : list of all the spatial codes as a single dataframe
- tab : the non squared input flow dataset with three column : origin, destination, flow value
- origin : node / place of origin of the flow
- dest : node / place of destination of the flow
- valflow : is the flow value between origin and destination places
- empty.sq : Builds an empty matrix or not. See Details.
- format : is the desired squared flow dataset output format. See Details.
- diagonale : to zero or not the main diagonal. See Details.

Details

- empty.sq is "TRUE" builds an empty matrix ; else is "FALSE" or missing
- format is "M" for matrice format
- format is "L" for long format, as three column dataframe
- diagonal is "TRUE" to zero the main diagonal
Examples

library(cartograflow)
data(flowdata)
var1 <- geoid
var2 <- flows

# 1/2 Compute an empty square matrice with ID code, and sets the value to zero
# Example for matrice format (same procedure for the long format)

mat <- flowcarre(var1, var2,
                  origin = "i", dest = "j", valflow = "Fij",
                  format = "M", empty.sq = TRUE
              )

# 2/2 Fill in the matrice with external flow values
mat <- flowcarre(var1, var2,
                  origin = "i", dest = "j", valflow = "Fij",
                  format = "M", empty.sq = FALSE
              )

# Square a matrice and zero the main diagonal
mat <- flowcarre(var1, var2,
                  origin = "i", dest = "j", valflow = "Fij",
                  format = "M", empty.sq = FALSE, diagonale = FALSE
              )

flowcontig                  Builds an ordinal distance matrices from a spatial features background

Description

From a layer of areal spatial features, compute an ordinal distance matrice based on a k order criterion of adjacency or contiguity between origin and destination places.
The result is a neighbourhood graph that can be used for filtering flow values before flow mapping (flowmap)

Usage

flowcontig(bkg, code, k, algo)

Arguments

bkg          a layer of areal spatial features (eg. the map background)
code         spatial areal features code
k            order of adjacency or contiguity between two areal spatial features
algo         algorithm to use for ordinal distance calculation. Default is "Dijkstra's" algorithm. See Details.
Details

The (k=1,2,...,k) order of adjacency or contiguity, of an areal spatial features background, is the number of spatial boundaries to be crossed between a couple of origin-destination (ODs) places. The k number can be assimilated to a shortest path between two pair of nodes. Argument ‘k’ is to enter the number k of the contiguity matrix to be constructed:
- kordre=1 : ODs places are adjacent, ie the flow have to cross only 1 boundary.
- kordre=2 : ODs places are distant from 2 borders
- kordre=k : ODs places are distant from k borders
The function returns also the (k) number of the layer

Value

a contiguity matrix with the k orders of adjacency

Examples

library(cartograflow)
library(sf)
data(flowdata)
map <- st_read(system.file("shape/MGP_TER.shp", package = "cartograflow"))
graph_ckij_1 <- flowcontig(bkg = map, code = "EPT_NUM", k = 1, algo = "automatic")
flowmap(
    tab = graph_ckij_1,
    fij = "ordre", origin.f = "i", destination.f = "j",
    bkg = map, code = "EPT_NUM", nodes.X = "X", nodes.Y = "Y",
    filter = FALSE
)

flowdist

Builds a continuous distance matrices from a spatial features background

Description

From a layer of areal spatial features, compute and threshold a continuous distance matrix. The result is either a matrice of distances between ODs, or a flow matrix based on the distance travelled between ODs; both can be used for filtering flow before flow mapping (flowmap)

Usage

flowdist(tab, dist.method, result)

Arguments

tab the input flow dataset

dist.method distance calculation algorithm, default is euclidian calculation

result Choose Building a "flowdist" or a simple "dist" matrice. See Details
Details

– result = "dist" is the simple resulting distance matrice.
– result = "flowdist" is the resulting distance matrice with additional calculated parameters.
– It is also possible to filter flow by a level of distance travelled.

Value

(1) A flowdata set with continuous distances calculations. See dist.method parameter
(2) A flowdata set with movement from euclidian distances calculations

Examples

library(cartograflow)
library(sf)
data(flowdata)
map <- st_read(system.file("shape/MGP_TER.shp", package = "cartograflow"))
tabflow <- flowjointure(
  geom = "area", bkg = map, DF.flow = flows, origin = "i",
  destination = "j", id = "EPT_NUM", x = "X", y = "Y"
)

# Format long with only origin, destination and distance parameters:
tab.distance <- flowdist(tabflow, dist.method = "euclidian", result = "dist")
# Format long with all parameters: coordinates, distance, mouvement

flowgini

Analysis of flow concentration (Gini coefficient)

Description

Calculates Gini coefficient, plot Lorenz curve and threshold the matrice according to a global concentration criterion for mapping flow intensity or flow density.
To be use before flowanalysis

Usage

flowgini(ODpts, origin, destination, valflow, lorenz.plot)

Arguments

ODpts the input dataset with : nodes code, flow values and XY coordinates
origin ID origin place, in long format
destination ID destination place, long format
valflow flow value between origin and destination places
lorenz.plot to plot or the Lorenz curve. See Details
Details

flowgini(...,lorenz.plot = TRUE) for ploting Lorenz curve associate to the gini coefficient, from cumulated flows and links.

Value

plot Lorenz curve for the cumulated flow and links : flowgini(...,gini.plot = TRUE),warning : the function must be not assign a variable

value of the Gini’s coefficient and the table : table<-flowgini(...,missing(gini.plot) or gini.plot = FALSE )

References


Examples

library(cartograflow)
data(flowdata)
# Computes Gini's coefficient
tabgini <- flowgini(ODpts = flows, origin = "i", destination = "j", valflow = "Fij", lorenz.plot = FALSE)
# Plot Lorenz curve
flowgini(ODpts = flows, origin = "i", dest = "j", valflow = "Fij", lorenz.plot = TRUE)
# See \link{flowanalysis} for viewing the tab_gini table

flowjointure

Builds a spatial join with a flow dataset

Description

Builds a spatial join between a flow dataset and a spatial features layer (as a map background)

Usage

flowjointure(geom, bkg, DF.flow, origin, destination, DF.point, id, x, y)

Arguments

gem | the geometry of the spatial features layer: points or areas
bkg | the spatial features layer
DF.flow | the input flow dataset as a dataframe
origin | the place of origin code
destination: the place of destination code
DF.point: a dataframe of points or places
id: dataframe of points or places file code
x: the X coordinate of the point or places
y: the Y coordinate of the point or places

Value
the corresponding joint table between the flow dataset and the spatial feature layer

Examples
library(cartograflow)
library(sf)
data(flowdata)
map <- st_read(system.file("shape/MGP_TER.shp", package = "cartograflow"))
tabflow <- flowjointure(
  geom = "area", bkg = map, DF.flow = flows, origin = "i", destination = "j",
  id = "EPT_NUM", x = "X", y = "Y"
)

---
flowlowup

Extracts the triangular sub-matrix of flows

Description
Extracts the upper or lower triangular part of a matrix

Usage
flowlowup(tab, origin = NULL, destination = NULL, fij = NULL, lowup, format, x)

Arguments
- tab: is the input flow dataset
- origin: the place of origin code
- destination: the place of destination code
- fij: the flow value between origin and destination places
- lowup: for selecting lower or upper triangular sub-portion of the original matrix. See Details.
- format: specify the flow dataset format, "M" for square matrix [n*n] or "L" for long [i,j,data]
- x: enter the triangular part to be extracted: "low", "up". See Details.
Details

This function computes for all pairs of origin-destination places (i,j) a lower "low" or upper "up" triangular sub-portion of the original matrix - x = "up" for the part above the main diagonal - x = "low" for the part below the main diagonal

Examples

library(cartograflow)
data(flowdata)

### Extract the upper part of the matrix : Long format

```r
tab_up <- flowlowup(flows, format="L", lowup="up")
tab_low <- flowlowup(flows, format="L", lowup="low")
```

---

**flowmap**

*Mapping of an origin-destination flow matrix*

Description

Mapping of an origin-destination flow matrix

Usage

```r
flowmap(
  tab, 
  fij, 
  origin.f, 
  destination.f, 
  bkg = NULL, 
  crs, 
  nodes = NULL, 
  code, 
  nodes.X, 
  nodes.Y, 
  filter, 
  plota, 
  threshold, 
  taille, 
  a.head, 
  a.length, 
  a.angle, 
  a.col, 
  add = NULL,
  ...
)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tab</td>
<td>the input flow dataset in .csv format. See Details</td>
</tr>
<tr>
<td>fij</td>
<td>the flow value between origin and destination places</td>
</tr>
<tr>
<td>origin.f</td>
<td>the place of origin code</td>
</tr>
<tr>
<td>destination.f</td>
<td>the place of destination code</td>
</tr>
<tr>
<td>bkg</td>
<td>a spatial feature layer, as a map background, in .shp or .json or other format</td>
</tr>
<tr>
<td>crs</td>
<td>the coordinate reference system (CRS)</td>
</tr>
<tr>
<td>nodes</td>
<td>the input points file in .csv format</td>
</tr>
<tr>
<td>code</td>
<td>the spatial features code</td>
</tr>
<tr>
<td>nodes.X</td>
<td>the X coordinate of the point or places</td>
</tr>
<tr>
<td>nodes.Y</td>
<td>the Y coordinate of the point or places</td>
</tr>
<tr>
<td>filter</td>
<td>is to filter or not the flow values. See details</td>
</tr>
<tr>
<td>plota</td>
<td>is to add spatial features as map background to the flows’s plot</td>
</tr>
<tr>
<td>threshold</td>
<td>the value of the threshold criterion to filter flows. Default is 1.</td>
</tr>
<tr>
<td>taille</td>
<td>the value of the width of the flow feature</td>
</tr>
<tr>
<td>a.head</td>
<td>for arrow’s head is the arrow head parameter code. It allows to choose the kind of arrow. See Details</td>
</tr>
<tr>
<td>a.length</td>
<td>for arrow’s length is the length of the edges of the arrow head (in inches)</td>
</tr>
<tr>
<td>a.angle</td>
<td>for arrow’s angle is the angle from the shaft of the arrow to the edge of the arrow head</td>
</tr>
<tr>
<td>a.col</td>
<td>for arrow’s color</td>
</tr>
<tr>
<td>add</td>
<td>is to allow to overlay flow features on external spatial features background</td>
</tr>
<tr>
<td>...</td>
<td>Adds the set of variables of the arrow function</td>
</tr>
</tbody>
</table>

Details

The input .csv flow dataset must be first converted to a dataframe for optimal performance (troubles remains with tibble format)

- filter is "FALSE" means that all the flow value will be plot as segments \([n*(n-1)]\), i.e. all the OD matrix’s cells out of the main diagonal will be plot.
- filter is "TRUE" means only non-zero values will be plot, i.e. existing links with or without threshold.

The default threshold is set to 1.

Flow features are plot as segments between \((x_0,y_0)\) and \((x_1,y_1)\)
- a.head is for applying an arrow or not to a segment:
  - code="0" : the link has no head - no arrow
  - code="1" : an arrow is draw at \((x_0[i], y_0[i])\)
  - code="2" : an arrow is draw at \((x_1[j], y_1[j])\)
  - code="3" : an arrow is draw at both nodes.
flowplaces

Description

Compute indicators per places (origin and/or destination) from the margins of the matrix. Ex/in and out degrees, gross and net flows, asymmetry... from an initial matrix.

Usage

flowplaces(tab, origin = NULL, destination = NULL, fij = NULL, format, x)

Arguments

- tab: is the input flow dataset
- origin: the place of origin code
- destination: the place of destination code
- fij: the flow value between origin and destination places
**flowreduct**

Flow matrix reduction according to another matrix

### Description

Reduces a flow dataset according to an external matrix, eg. a matrix of travelled distance. Builds geographical movements, by weighting a flow dataset according to a distance criterion.

### Usage

```r
flowreduct(tab, tab.metric, metric, d.criteria, d)
```

---

**format**

specify the flow dataset format, "M" for square matrix [n*n] or "L" for long [i,j,data]

**x**

type the flowplaces indicator type : "allflowplaces", "ini", "outi", "degi", "intra", "Oi", "Dj", voli", "bali", "asyi". See Details.

**Examples**

```r
library(cartograflow)
data(flowdata)
bkg <- system.file("shape/MGP_TER.shp", package = "cartograflow", lib.loc = NULL, mustWork = TRUE)

###1: Computes the total flow volume of places : Long format
voli <- flowplaces(flows, origin = "i", destination = "j", fij = "Fij", format = "L", x = "voli")

###2: Computes the total flows received by destination place : Long format
tab_bali <- flowplaces(flows, origin = "i", destination = "j", fij = "Fij", format = "L", x = "bali")
```
Arguments

- `tab` is the input flow data set
- `tab.metric` is the distance dataset
- `metric` select "continuous" or "ordinal" metric. See Details
- `d.criteria` is for selecting "dmin" or "dmax" distance criteria for "continuous" metric. See Details.
- `d` is the value of the selected "dmin" or "dmax". See Details

Details

The involved metric can be continuous or not.

(1) Metric is 'continuous' for distance as euclidian, maximum, manhattan, etc. See `flowdist`
- Metric is 'ordinal' for computing neighbourhood ordinal distance matrix. – Select ="dmin" for reducing flow dataset to flow values that are up or equal to the dmin distance parameter (Fij>=dmin);
- select ="dmax" for reducing flow dataset to values that are less or equal to the dmax distance parameter(Fij=<dmax).

- Metric is 'ordinal' for computing neighbourhood ordinal distance with k contiguity. See `flowcontig` for computing ordinal distance matrix

Value

A flow dataset with distances computations and flow reduction

Examples

```r
library(cartograflow)
library(sf)
library(dplyr)
data(flowdata)
map <- st_read(system.file("shape/MGP_TER.shp", package = "cartograflow"))

tab <- flowjointure(
  geom = "area", bkg = map, DF.flow = flows, origin = "i", destination = "j",
  id = "EPT_NUM", x = "X", y = "Y"
)

# Example for reducing a flow matrice with a distance matrice, in long format (i,j, distance)
## 1/2: Computes the matrice distances
tab.distance <- flowdist(tab, dist.method = "euclidian", result = "dist")
## 2/2: Reduce the flow matrice
# library(dplyr)
# tab.flow <- tab %>% mutate(flowreduct(flows, tab.distance, metric = "continous"))
```
flows

\[
d.\text{criteria} = "dmax", \ d = 8567
\]

---

**flows**  
**MOBPRO: Commuting trips in 2015**

**Description**

Citation: INSEE - RP MOBPRO, 2015.  
Variable (i) is the place of origin of the flow.  
Variable (j) is the place of destination of the flow.  
Variable (Fij) is the flow value between (i,j).  
Variable (count) is the frequency of the (i,j) couple of places.  

**Source**  
[https://www.insee.fr/fr/statistiques/fichier/3566008/rp2015_mobpro_txt.zip](https://www.insee.fr/fr/statistiques/fichier/3566008/rp2015_mobpro_txt.zip)

---

**flowstructmat**  
**Structuring a matrix**

**Description**

Fixes an ID shift in the flow matrix (to be used with flowjointure if necessary and flowtabmat)

**Usage**

```r
flowstructmat(z)
```

**Arguments**

- `z` The input flow dataset in the matrice format where the first column is filled with the ID

**Value**

A flow dataset with an usable format
Examples

```r
library(cartograflow)
data(flowdata)

dim(mat_ex) # dimension of the original matrix
### 10 11 # first column is filled with the ID

tab <- flowstructmat(mat_ex)
dim(tab)
## 10 10 # dimension of the resulting matrix
```

### flowtabmat

*Changing the format of a flow dataset*

**Description**

Transform a flow dataset from long to matrix format, and vice versa. Square matrix.

**Usage**

`flowtabmat(tab, matlist)`

**Arguments**

- `tab`: flow dataset, in matrix or long format
- `matlist`: choose "matrix" or "long" for the resulting format. See Details.

**Details**

- matlist="M" from long (3 columns: origin, destination, flow) to matrix format [n*n];
- matlist="L" from matrix to long format.

**Value**

A flow dataset in matrix or in long format

**Examples**

```r
library(cartograflow)
data(flowdata)

# 1: From long to matrix format (n*m)
matFlow <- flowtabmat(flows, matlist = "M")

# 2: From matrix to long format [i,j,Fij]
listflow <- flowtabmat(matFlow, matlist = "L")
```
flowtype

Compute bilateral several flow types

Description
Compute bilateral flow type: volume (gross), balance (net), asymmetry, min/max ... from an initial asymmetric matrix

Usage
flowtype(
  tab,
  origin = NULL,
  destination = NULL,
  fij = NULL,
  format,
  lowup,
  net,
  x
)

Arguments
- **tab**: is the input flow dataset
- **origin**: the place of origin code
- **destination**: the place of destination code
- **fij**: the flow value between origin and destination places
- **format**: specify the flow dataset format, "M" for square matrix [n*n] or L for long [i,j,data]
- **lowup**: for extracting the lower or upper triangular sub-portion of the bilateral volume matrix. See Details.
- **net**: for extracting the "positive" or the "negative" flow values of the bilateral balance matrix
- **x**: enter the flow indicator type: "alltypes", "flux", "transpose", "bivolum", "bibal", "biasym", "bimin", "bimax", "birange" and "bidisym"

Details
The matrix must be squared (if not, see flowcarre). This function compute for all pairs or origin-destination places (i,j) involved in an asymmetric flow matrix (Fij<> Fji) several matrix:

- Param x: - x = "flux" for the initial flow: (Fij)
- x = "transpose" for the reverse flow value: (Fji) = t(Fij)
- x = "bivolum" for the bilateral volume or gross flow: FSij = (Fij+Fji)
- x = "bibal" for the bilateral balance or net flow: FBij = (Fij-Fji)
- x = "biasym" for asymmetry of bilateral flow: FAij = (FBij/FSij)
- \( x = \) "bimin" for the minimum of bilateral flow: \( \min F_{ij} = (F_{ij}, F_{ji}) \)
- \( x = \) "bimax" for the maximum of bilateral flow: \( F_{ij}(F_{ij}, F_{ji}) \)
- \( x = \) "birange" for the amplitude of bilateral flows: \( \text{range} F_{ij} = (\max F_{ij} - \min F_{ij}) \)
- \( x = \) "bidisym" for the bilateral disymetry: \( \text{FD}_{ij} = (\text{range} F_{ij}/\text{FS}_{ij}) \)
- \( x = \) "alltypes" for computing all the available types of flows

Param lowup is for reducing the matrix:
- lowup = "up" for triangular part above the main diagonal
- lowup = "low" for triangular part below the main diagonal

Param net is for extracting positive or negative flow values of the bilateral balance (bibal matrix):
- net = "negative" values
- net = "positive" values

References


Examples

```r
library(cartograflow)
data(flowdata)
bkg <- system.file("shape/MGP_TER.shp",
    package = "cartograflow",
    lib.loc = NULL, mustWork = TRUE
)
#
## 1a: Computes flowtypes: Matrix format
matflow <- flowtabmat(flows, matlist = "M")
m <- flowtype(matflow, format = "M", x = "flux")
m <- flowtype(matflow, format = "M", x = "transpose")
m <- flowtype(matflow, format = "M", x = "bivolum")
m <- flowtype(matflow, format = "M", x = "bibal")

## 1b: Computes flowtypes: Long format
types_all <- flowtype(flows, origin = "i", destination = "j", fij = "Fij", format = "L",
x = "alltypes")
bivol <- flowtype(flows, origin = "i", destination = "j", fij = "Fij", format = "L",
x = "bivolum", lowup = "up")
bibal_net <- flowtype(flows, origin = "i", destination = "j", fij = "Fij", format = "L",
x = "bibal", net = "negative")

# 2: Flowmapping: example of bibal_net
flowmap(bibal_net,
    format = "L", bkg, code = "EPT_NUM",
    filter = TRUE, threshold = 20, taille = 5, a.head = 1,
)
```
<table>
<thead>
<tr>
<th>geoid</th>
<th>Geographical ID</th>
</tr>
</thead>
</table>

**Description**

One column dataframe in.csv.
Variable (COD_GEO_EPT) is the geographical code of the territory
Citation: APUR, 2018

**Source**

https://www.insee.fr/fr/statistiques/fichier/3566008/rp2015_mobpro_txt.zip

<table>
<thead>
<tr>
<th>mat_ex</th>
<th>Example of a small matrix</th>
</tr>
</thead>
</table>

**Description**

Example of a small Origin-Destination flow dataset, in a matriz format
Index

cartograflow, 2
flowanalysis, 2, 2, 7
flowcarre, 2, 4, 17
flowcontig, 2, 5, 14
flowdist, 2, 6, 14
flowgini, 2, 3, 7
flowjointure, 2, 8, 15
flowlowup, 9
flowmap, 2, 5, 6, 10
flowplaces, 12
flowreduct, 2, 13
flows, 15
flowstructmat, 2, 15
flowtabmat, 13, 16
flowtype, 17

geoid, 19

mat_ex, 19