Package 'IGC.CSM'

March 4, 2026

Title Simulate Impact of Different Urban Policies Through a General
Equilibrium Model

Version 0.3.1

Description Develops a General Equilibrium (GE) Model, which estimates key vari-
ables such as wages, the number of residents and work-
ers, the prices of the floor space, and its distribution between commercial and residen-
tial use, as in Ahlfeldt et al., (2015) <doi:10.3982/ECTA10876>. By doing so, the model al-
lows understanding the economic influence of different urban policies.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.1

Author David Zarruk [aut],
Roman Zarate [aut, cre]

Maintainer Roman Zarate <rd.zarate40@gmail.com>

NeedsCompilation no

Repository CRAN

Date/Publication 2026-03-04 17:20:10 UTC

Contents

array_operator	*Array operator to mimic different-dimension-array element-wise operations in MATLAB. It receives as input two arrays of potentially different dimensions, it resizes them to have same dimensions and finally performs the element-wise operation.*

Description

Array operator to mimic different-dimension-array element-wise operations in MATLAB. It receives as input two arrays of potentially different dimensions, it resizes them to have same dimensions and finally performs the element-wise operation.

Usage

```
array_operator(array1, array2, operation)
```

Arguments

array1	The first array
array2	The second array
operation	The operation. It can take values: '+', '-', '*', '/' and '^'

Value

An array with dimensions equal to the "largest" input array. It is the result of applying the operator element-wise to both input arrays.

av_income_simple	*Computes average income in each location, which is the weighted average of the income of the people living in the location.*

Description

Computes average income in each location, which is the weighted average of the income of the people living in the location.

Usage

```
av_income_simple(lambda_ij_i, w_tr)
```

Arguments

lambda_ij_i	NxN matrix - Probability of individuals in each location of working in each location.
w_tr	NxS - Wages in each location in each sector.

---

commuting_matrix	*Function to transform travel times into iceberg commuting costs*

Description

Function to transform travel times into iceberg commuting costs

Usage

```
commuting_matrix(t_ij, epsilon)
```

Arguments

t_ij	NxN matrix - Travel time matrix across locations
epsilon	Float - Parameter that transforms travel times to commuting costs

Value

A NxN matrix of commuting costs

---

density_development	*Computes residential and commercial floorspace supply and equilibrium prices.*

Description

Computes residential and commercial floorspace supply and equilibrium prices.

Usage

```
density_development(Q, K, w, L_j, y_bar, L_i, beta, alpha, mu)
```

Arguments

Q	Nx1 array - Floorspaces prices.
K	Nx1 array - Land supply.
w	NxS - Wages in each location in each sector.
L_j	Nx1 matrix - Number of workers in each location.
y_bar	• Average income in each location.
L_i	Nx1 matrix - Number of residents in each location.
beta	Float - Cobb-Douglas parameter output elasticity wrt labor.
alpha	Float - Utility parameter that determines preferences for consumption.
mu	Float - Floorspace prod function: output elast wrt capita, 1-mu wrt land.

---

inversionModel   *Function to invert model, so amenities, wages, productivities, and de-*
                        velopment density are chosen to match model to data.

Description

Function to invert model, so amenities, wages, productivities, and development density are chosen
to match model to data.

Usage

```
inversionModel(
  N,
  L_i,
  L_j,
  Q,
  K,
  t_ij,
  alpha = 0.7,
  beta = 0.7,
  theta = 7,
  delta = 0.3585,
  rho = 0.9094,
  lambda = 0.01,
  epsilon = 0.01,
  mu = 0.3,
  eta = 0.1548,
  nu_init = 0.05,
  tol = 10^-10,
  maxiter = 1000,
  verbose = FALSE
)
```

Arguments

N	Integer - Number of locations.
L_i	Nx1 matrix - Number of residents in each location.
L_j	Nx1 matrix - Number of workers in each location.
Q	Nx1 matrix - Floorspace prices
K	Nx1 matrix - Land area
t_ij	NxN matrix - Travel times across all possible locations.
alpha	Float - Utility parameter that determines preferences for consumption.
beta	Float - Output elasticity wrt labor
theta	Float - Commuting elasticity and migration elasticity.

delta	Float - Decay parameter agglomeration
rho	Float - Decay parameter congestion
lambda	Float - Agglomeration force
epsilon	Float - Parameter that transforms travel times to commuting costs
mu	Float - Floorspace prod function: output elast wrt capital, 1-mu wrt land.
eta	Float - Congestion force
nu_init	Float - Convergence parameter to update wages. Default nu=0.01.
tol	Int - tolerance factor
maxiter	Integer - Maximum number of iterations for convergence. Default maxiter=1000.
verbose	Boolean - Equal to TRUE to print verbose.

Value

A list containing equilibrium values: population (L_i), employment (L_j), productivities (A, a), real income (u), amenities (B, b), wages (w), density of development (varphi), welfare (U), normalized prices (Q_norm), and commercial floorspace share (ttheta).

Examples

```
N=5
L_i = c(63, 261, 213, 182, 113)
L_j = c(86, 278, 189, 180, 99)
Q = c(2123, 1576, 1371, 1931, 1637)
K = c(0.44, 1.45, 1.15, 0.87, 0.58)
t_ij = rbind(c(0.0, 6.6, 5.5, 5.6, 6.4),
             c(6.7, 0.0, 3.9, 4.6, 4.4),
             c(5.5, 3.9, 0.0, 2.8, 3.0),
             c(5.6, 4.6, 2.8, 0.0, 2.7),
             c(6.4, 4.4, 3.0, 2.7, 0.0))

inversionModel(N=N,
               L_i=L_i,
               L_j=L_j,
               Q=Q,
               K=K,
               t_ij=t_ij)
```

---

living_amenities_simple

Function to estimate amenity parameters of locations where users live.

Description

Function to estimate amenity parameters of locations where users live.

Usage

```
living_amenities_simple(theta, N, L_i, W_i, Q, K, alpha, t_ij, rho, eta)
```

Arguments

theta	Float - Parameter that governs the reallocation of workers across locations in the city. This parameter measures how sensible are migration flows within the city to changes in real income.
N	Integer - Number of locations.
L_i	Nx1 matrix - Total residents.
W_i	Nx1 matrix - Market access measure in each location.
Q	Nx1 matrix - Floor space prices.
K	Nx1 matrix - Land area
alpha	Float - Para
t_ij	NxN matrix - Travel times across locations.
rho	Float - decay parameter for amenities.
eta	Float - congestion force

Value

Matrix with the amenity distribution of living in each location.

NoPricesInversionModel

> *Function to invert model, so amenities, wages, productivities, and development density are chosen to match model to data. This function requires data on the amount of floorspace.*

Description

Function to invert model, so amenities, wages, productivities, and development density are chosen to match model to data. This function requires data on the amount of floorspace.

Usage

```
NoPricesInversionModel(
  N,
  L_i,
  L_j,
  K,
  t_ij,
  alpha = 0.7,
  beta = 0.7,
  theta = 7,
```

```
    delta = 0.3585,
    rho = 0.9094,
    lambda = 0.01,
    epsilon = 0.01,
    mu = 0.3,
    eta = 0.15,
    nu_init = 0.05,
    tol = 10^-10,
    maxiter = 1000,
    FS
)
```

Arguments

N	Integer - Number of locations.
L_i	Nx1 matrix - Number of residents in each location.
L_j	Nx1 matrix - Number of workers in each location.
K	Nx1 matrix - Land area
t_ij	NxN matrix - Travel times across all possible locations.
alpha	Float - Utility parameter that determines preferences for consumption.
beta	Float - Output elasticity wrt labor
theta	Float - Commuting elasticity and migration elasticity.
delta	Float - Decay parameter agglomeration
rho	Float - Decay parameter congestion
lambda	Float - Agglomeration force
epsilon	Float - Parameter that transforms travel times to commuting costs
mu	Float - Floorspace prod function: output elasticity wrt capital, 1-mu wrt land.
eta	Float - Congestion force
nu_init	Float - Convergence parameter to update wages. Default nu=0.01.
tol	Int - tolerance factor
maxiter	Integer - Maximum number of iterations for convergence. Default maxiter=1000.
FS	Nx1 array - Total floorspace in the location

Value

A list containing the equilibrium values for the variables of the model. The reported variables are population (Li), employment (Lj), normalized rents (Q_norm), endogenous amenities (A), exogenous amenities (a), endogenous productivities (B), exogenous productivities (b), wages (w), density of development (varphi), aggregate welfare (U), share of commercial floorspace (ttheta), floorspace (FS), residential floorspace (FS_r) and commercial floorspace (FS_f)

PricesInversionModel   *Function to invert model, so amenities, wages, productivities, and de-*
                              velopment density are chosen to match model to data. In this version
                              you require housing prices

Description

Function to invert model, so amenities, wages, productivities, and development density are chosen to match model to data.

Usage

```
PricesInversionModel(
  N,
  L_i,
  L_j,
  Q,
  K,
  t_ij,
  alpha = 0.7,
  beta = 0.7,
  theta = 7,
  delta = 0.3585,
  rho = 0.9094,
  lambda = 0.01,
  epsilon = 0.01,
  mu = 0.3,
  eta = 0.1548,
  nu_init = 0.05,
  tol = 10^-10,
  maxiter = 1000
)
```

Arguments

N	Integer - Number of locations.
L_i	Nx1 matrix - Number of residents in each location.
L_j	Nx1 matrix - Number of workers in each location.
Q	Nx1 matrix - Floorspace prices
K	Nx1 matrix - Land area
t_ij	NxN matrix - Travel times across all possible locations.
alpha	Float - Utility parameter that determines preferences for consumption.
beta	Float - Output elasticity wrt labor
theta	Float - Commuting elasticity and migration elasticity.

delta	Float - Decay parameter agglomeration
rho	Float - Decay parameter congestion
lambda	Float - Agglomeration force
epsilon	Float - Parameter that transforms travel times to commuting costs
mu	Float - Floorspace prod function: output elast wrt capital, 1-mu wrt land.
eta	Float - Congestion force
nu_init	Float - Convergence parameter to update wages. Default nu=0.01.
tol	Int - tolerance factor
maxiter	Integer - Maximum number of iterations for convergence. Default maxiter=1000.

Value

A list containing equilibrium values: population (L_i), employment (L_j), productivities (A, a), real income (u), amenities (B, b), wages (w), density of development (varphi), welfare (U), normalized prices (Q_norm), commercial floorspace share (ttheta), total floorspace (FS), residential floorspace (FS_r), and commercial floorspace (FS_f).

Examples

```
N=5
L_i = c(63, 261, 213, 182, 113)
L_j = c(86, 278, 189, 180, 99)
Q = c(2123, 1576, 1371, 1931, 1637)
K = c(0.44, 1.45, 1.15, 0.87, 0.58)
t_ij = rbind(c(0.0, 6.6, 5.5, 5.6, 6.4),
             c(6.7, 0.0, 3.9, 4.6, 4.4),
             c(5.5, 3.9, 0.0, 2.8, 3.0),
             c(5.6, 4.6, 2.8, 0.0, 2.7),
             c(6.4, 4.4, 3.0, 2.7, 0.0))

PricesInversionModel(N=N,
                     L_i=L_i,
                     L_j=L_j,
                     Q=Q,
                     K=K,
                     t_ij=t_ij)
```

---

productivity	*Computes productivity levels in each location*

Description

Computes productivity levels in each location

Usage

```
productivity(N, Q, w, L_j, K, t_ij, delta, lambda, beta)
```

Arguments

N	Float - Number of locations.
Q	Nx1 matrix - Floorspace prices in each location.
w	Nx1 matrix - wages in each location.
L_j	Nx1 matrix - Employment in each location.
K	Nx1 matrix - Land in each location.
t_ij	NxN matrix - Travel times matrix.
delta	Float - decay parameter agglomeration.
lambda	Float - agglomeration force.
beta	Float - Output elasticity wrt labor

---

solveModel   *Function to solve counterfactuals.*

Description

Function to solve counterfactuals.

Usage

```
solveModel(
  N,
  L_i,
  L_j,
  K,
  t_ij,
  a,
  b,
  varphi,
  w_eq,
  u_eq,
  Q_eq,
  ttheta_eq,
  alpha = 0.7,
  beta = 0.7,
  theta = 7,
  mu = 0.3,
  delta = 0.3585,
  lambda = 0.01,
  rho = 0.9094,
  eta = 0.1548,
  epsilon = 0.01,
  zeta = 0.95,
  varrho = 3,
```

```
    sh_city = 0.1,
    tol = 10^-10,
    maxiter = 1000,
    verbose = FALSE
)
```

Arguments

N	Integer - Number of locations.
L_i	Nx1 array - Number of residents in each location
L_j	Nx1 array - Number of workers in each location
K	Nx1 array - Land supply
t_ij	NxN matrix - Travel times across locations
a	Nx1 array - Total Factor Productivity in each location
b	Nx1 array - Vector of amenities in each location
varphi	Nx1 array - Density of development
w_eq	Nx1 array - Initial vector of wages
u_eq	Nx1 array - Initial vector of real income
Q_eq	Nx1 array - Initial price for floorspace
ttheta_eq	Nx1 array - Share of floorspace used commercially
alpha	Float - Exp. share in consumption, 1-alpha exp. share in housing
beta	Float - Output elasticity with respect to labor
theta	Float - Commuting and migration elasticity.
mu	Float - Floorspace prod function: output elasticity wrt capital
delta	Float - Decay parameter agglomeration force
lambda	Float - agglomeration externality
rho	Float - decay parameter for amenities
eta	Float - amenity externality
epsilon	Float - Parameter that transforms travel times to commuting costs
zeta	Float - convergence parameter
varrho	Float - migration elasticty
sh_city	Share of population living in the city
tol	Float - tolerance factor
maxiter	Integer - Maximum number of iterations for convergence. Default maxiter=1000.
verbose	Boolean - Equal to TRUE to print verbose.

Value

A list containing new equilibrium values for the variables of the model. The reported variables are wages (w), residents market access (W_i), endogenous productivities (A), endogenous amenities (B), population (Li), employment (Lj), share of commercial floorspace (ttheta), real income (u), welfare (U), firms market access (FCMA), firms output (Y).

---

sumDims   *Collapse array along one of the dimensions by adding the elements along that dimension.*

Description

Collapse array along one of the dimensions by adding the elements along that dimension.

Usage

```
sumDims(array, dimension)
```

Arguments

array   Array to collapse along one dimension.

dimension   Dimension to collapse the array.

Value

An array that has been collapsed along the given dimension.

---

sumDims2   *Collapse array 2 along one of the dimensions by adding the elements along that dimension.*

Description

Collapse array 2 along one of the dimensions by adding the elements along that dimension.

Usage

```
sumDims2(array, dimension)
```

Arguments

array   Array to collapse along one dimension.

dimension   Dimension to collapse the array.

Value

An array that has been collapsed along the given dimension.

wages_inversion	*Function to compute equilibrium wages that make the model labor in every location in equal to the observed data. It finds the w's such that equation (3.2) holds.*

Description

Function to compute equilibrium wages that make the model labor in every location in equal to the observed data. It finds the w's such that equation (3.2) holds.

Usage

```
wages_inversion(
  N,
  w_init,
  theta,
  tau,
  L_i,
  L_j,
  nu_init = 0.05,
  tol = 10^-10,
  maxiter = 10000,
  verbose = FALSE
)
```

Arguments

N	Integer - Number of locations.
w_init	Initial vector of wages.
theta	Float - Commuting elasticity.
tau	NxN matrix - Commuting cost matrix across all locations.
L_i	Nx1 matrix - Number of residents in each location.
L_j	Nx1 matrix - Number of workers in each location.
nu_init	Float - Convergence parameter to update wages. Default nu=0.01.
tol	Float - Maximum tolerable error for estimating total labor. Default tol=10^-10.
maxiter	Integer - Maximum number of iterations for convergence. Default maxiter=10000.
verbose	Boolean - Equal to TRUE to print verbose.

Value

A list with equilibrium wages and probability of workers in each location working in every other location.

Index