Package ‘drSpaceTime’

April 6, 2016

Title Spatial-Temporal data analysis using divide and recombined

Version 1.0

Description Apply loess smoothing to spatial-temporal data using divide and recombined concept. Divide the data either by time or by location, and then apply either spatial loess smoothing fit to the by time division or stlplus fit to the by location division.

Date 2016-04-06

Depends R (>= 3.1.1)

Imports Spaloess,
roxygen2,
plyr,
Rcpp,
yaImpute,
stlplus

License MIT

LazyData true

RoxygenNote 5.0.1

Suggests knitr,
rmarkdown

VignetteBuilder knitr

LinkingTo Rcpp

R topics documented:

mapreduce.control *Set mapreduce parameter for drSpaceTime fitting*

Description

Set control parameters of mapreduce for drSpaceTime fits

Usage

```
mapreduce.control(reduceTask = 0, libLoc = NULL, BLK = 512,
  map_jvm = "-Xmx3000m", reduce_jvm = "-Xmx4096m", map_memory = 5120,
  reduce_memory = 5120, slow_starts = 0.9, spill_percent = 0.8,
  io_sort = 100, task_io_sort_factor = 10, reduce_parallelcopies = 5,
  reduce_shuffle_input_buffer_percent = 0.7,
  reduce_shuffle_merge_percent = 0.66, reduce_merge_inmem = 1000,
  reduce_input_buffer_percent = 0, reduce_buffer_read = 150,
  map_buffer_read = 150, reduce_buffer_size = 10000,
  map_buffer_size = 10000)
```

Arguments

`reduceTask` The reduce task number, also the number of output files. If set to be 0, then there is no shuffle and sort stage after map.

`spill_percent` For mapreduce.sort.spill.percent parameter, the threshold usage proportion for both the map output memory buffer and record boundaries index to start the process of spilling to disk.

`io_sort` For mapreduce.task.io.sort.mb, the size, in megabytes, of the memory buffer to use while sorting map output.

`task_io_sort_factor`
For mapreduce.task.io.sort.factor

`reduce_parallelcopies`
For mapreduce.reduce.shuffle.parallelcopies

`reduce_shuffle_input_buffer_percent`
For mapreduce.reduce.shuffle.input.buffer.percent

`reduce_shuffle_merge_percent`
For mapreduce.reduce.shuffle.merge.percent

`reduce_merge_inmem`
For mapreduce.reduce.merge.inmem.shreshold

`reduce_input_buffer_percent`
For mapreduce.reduce.input.buffer.percent

Value

A list with mapreduce tuning parameters.

Author(s)

Xiaosu Tong

Examples

```
mapreduce.control()
```

readIn *Readin Raw text data files and save it as by time division on HDFS.*

Description

Input raw text data file is divided into by time subsets and saved on HDFS

Usage

```
readIn(input, output, info, cluster_control = mapreduce.control())
```

Arguments

`input` The path of input file on HDFS. It should be raw text file.

`output` The path of output file on HDFS. It is by time division.

`info` The RData path on HDFS which contains all station metadata

`cluster_control` all parameters that are needed for mapreduce job

Author(s)

Xiaosu Tong

Examples

```
FileInput <- "/wsc/tongx/spatem/nRaw/tmax"
FileOutput <- "/wsc/tongx/spatem/tmax/test/bymth"
ccontrol <- mapreduce.control(
  libLoc=lib.loc, reduceTask=179, io_sort=512, BLK=256, slow_starts = 0.8,
  reduce_input_buffer_percent=0.9, reduce_parallelcopies=5,
  reduce_merge_inmem=0, task_io_sort_factor=100,
  spill_percent=0.9, reduce_shuffle_input_buffer_percent = 0.9,
  reduce_shuffle_merge_percent = 0.5,
  reduce_buffer_read = 100, map_buffer_read = 100,
  reduce_buffer_size = 10000, map_buffer_size = 10000
)
readIn(
  FileInput, FileOutput, info="/hdfs/path/a1950UStinfo.RData",
  cluster_control=ccontrol
)
```

`spacetime.control` *Set smoothing parameters for drSpaceTime fitting*

Description

Set control parameters for the smoothing fit

Usage

```
spacetime.control(vari = "resp", time = "date", seaname = "month", n,
  n.p = 12, s.window, s.degree = 1, t.window = NULL, t.degree = 1,
  inner = 4, outer = 1, statbytime = 2, degree, span, Edeg,
  surf = c("interpolate", "direct"), iterations = 1)
```

Arguments

`vari` variable name in string of the response variable

`time` variable name in string of time index of the whole time series

`seaname` variable name in string of the seasonal variable

`n` the number of total observations

`n.p` the number of observation in each subseries

`s.window` either the character string `"periodic"` or the span (in lags) of the loess window for seasonal extraction, which should be odd. This has no default.

`s.degree` degree of locally-fitted polynomial in seasonal extraction. Should be 0, 1, or 2.

`inner` The iteration time for inner loop of stlplus for time dimension fitting

`outer` The iteration time for outer loop of stlplus for time dimension fitting

`statbytime` The number of locations will be grouped together in the by time division after stlfit. The parameter is only used for `swaptoTime`. Since there may be to many time point in each location, the `swaptoTime` looping all time point will add to much overhead caused by `rhcollect`. So every `statbytime` time point are collect into one key-value pair.

`degree` smoothing degree for the spatial loess smoothing. It can be 0, 1, or 2.

`span` smoothing span for the spatial loess smoothing.

`Edeg` the degree for the conditioanl parametric model including elevation.

`surf` should the fitted surface be computed exactly or via interpolation from a kd tree.

`iterations` the number of iterations used for the space-time back-fitting.

Value

A list with space-time fitting parameters.

Author(s)

Xiaosu Tong

References

R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, **6**, 3–73.

Examples

```
spacetime.control(
 n = 786432, n.p = 12, s.window = 21, s.degree = 1, t.window = 241,
  t.degree = 1, degree = 2, span = 0.015, Edeg = 2, surf = "interpolate"
)
```

---

spaofit *Conduct the spatial loess fitting on the original observations at each location in parallel*

Description

Call `spaloess` function on the spatial domain at each time point in parallel. Every spatial domain uses the same smoothing parameters. NA observations will be imputed.

Usage

```
spaofit(input, output, info, model_control = spacetime.control(),
  cluster_control = mapreduce.control())
```

Arguments

`input` The path of input sequence file on HDFS. It should be by location division.

`output` The path of output sequence file on HDFS. It is by location division but with seasonal and trend components

`info` The RData path on HDFS which contains all station metadata

`model_control` Should be a list object generated from `spacetime.control` function. The list including all necessary smoothing parameters of nonparametric fitting.

`cluster_control`
Should be a list object generated from `mapreduce.control` function. The list including all necessary Rhipe parameters and also user tunable MapReduce parameters.

Author(s)

Xiaosu Tong

Examples

```
FileInput <- "/wsc/tongx/spatem/tmax/sim/bymth"
FileOutput <- "/wsc/tongx/spatem/tmax/sim/bymthfit"
ccontrol <- mapreduce.control(
  libLoc=lib.loc, reduceTask=0, BLK=128, map_jvm = "-Xmx3584m",
  map_memory = 5120, map_buffer_read = 100, map_buffer_size = 1000
)
mcontrol <- spacetime.control(
  vari="resp", time="date", seaname="month", n=786432, n.p=12,
  s.window=13, t.window = 241, degree=2, span=0.015, Edeg=2
)
spaofit(
  FileInput, FileOutput, target=you$vari, na=TRUE,
  info="/wsc/tongx/spatem/stationinfo/a1950UStinfo.RData",
  model_control=mcontrol, cluster_control=ccontrol
)
```

`sparfit` *Conduct the spatial loess fitting on the remainder at each location in parallel*

Description

Call `spaloess` function on the spatial domain at each time point in parallel. Every spatial domain uses the same smoothing parameters

Usage

```
sparfit(input, output, info, model_control = spacetime.control(),
  cluster_control = mapreduce.control())
```

Arguments

`input` The path of input file on HDFS. It should be by location division.

`output` The path of output file on HDFS. It is by location division but with seasonal and trend components

`info` The RData path on HDFS which contains all station metadata

`model_control` Should be a list object generated from `spacetime.control` function. The list including all necessary smoothing parameters of nonparametric fitting.

`cluster_control` Should be a list object generated from `mapreduce.control` function. The list including all necessary Rhipe parameters and also user tunable MapReduce parameters.

Author(s)

Xiaosu Tong

Examples

```
FileInput <- "/wsc/tongx/spatem/tmax/sim/bymthse"
FileOutput <- "/wsc/tongx/spatem/tmax/sim/bymthfitse"
ccontrol <- mapreduce.control(
  libLoc=lib.loc, reduceTask=0, BLK=128, map_jvm = "-Xmx3584m",
  map_memory = 5120, map_buffer_read = 100, map_buffer_size = 1000
)
mcontrol <- spacetime.control(
  vari="remainder", time="date", seaname="month", n=786432, n.p=12,
  s.window=13, t.window = 241, degree=2, span=0.015, Edeg=2
)
sparfit(
  FileInput, FileOutput, target=you$vari, na=TRUE,
  info="/wsc/tongx/spatem/stationinfo/a1950UStinfo.RData",
  model_control=mcontrol, cluster_control=ccontrol
)
```

`stlfit` *Conduct the stlplus fitting at each location in parallel*

Description

Calling `stlplus` function from Ryan Hafen's `stlplus` package on time series at each location in parallel. Every station uses the same smoothing parameter

Usage

```
stlfit(input, output, model_control = spacetime.control(),
  cluster_control = mapreduce.control())
```

Arguments

`input` The path of input sequence file on HDFS. It should be by location division.

`output` The path of output sequence file on HDFS. It is by location division but with seasonal and trend components

`model_control` The list contains all smoothing parameters

`cluster_control` A list contains all mapreduce tuning parameters.

Author(s)

Xiaosu Tong

References

R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, **6**, 3–73.

Ryan Hafen (2010): stlplus: Local regression models: Advancements, applications, and new methods. *Purdue University*

See Also

spacetime.control, mapreduce.control

Examples

```
FileInput <- "/wsc/tongx/spatem/tmax/sim/bystat"
FileOutput <- "/wsc/tongx/spatem/tmax/sim/bystatfit"
ccontrol <- mapreduce.control(libLoc=lib.loc, reduceTask=0)
mcontrol <- spacetime.control(
  vari = "resp", time = "date", seaname = "month",
  n = 786432, n.p = 12, s.window = "periodic", t.window = 241,
  degree = 2, span = 0.015, Edeg = 2
)
stlfit(FileInput, FileOutput, model_control=mcontrol, cluster_control=ccontrol)
```

swaptoLoc *Swap to division by location*

Description

Switch input key-value pairs which is division by time to the key-value pairs which is division by location.

Usage

```
swaptoLoc(input, output, cluster_control = mapreduce.control())
```

Arguments

input	The path of input file on HDFS. It should be by location division.
output	The path of output file on HDFS. It is by time division.
cluster_control	Should be a list object generated from mapreduce.control function. The list including all necessary Rhipe parameters and also user tunable MapReduce parameters.

Details

The value of each input key-value pair is a vector of spatial smoothed value in the same order based on location index. For each of element of this vector a intermediate key-value pair is collected. The value of final output key-value pair is a vector in order to keep the size as small as possible, and also guarantee the combiner can set to be TRUE.

Author(s)

Xiaosu Tong

See Also

spacetime.control, mapreduce.control

Examples

```
FileInput <- "/wsc/tongx/spatem/tmax/sim/bymthfit"
FileOutput <- "/wsc/tongx/spatem/tmax/sim/bystat"
ccontrol <- mapreduce.control(
  libLoc=lib.loc, reduceTask=169, io_sort=768, BLK=256, slow_starts = 0.5,
  map_jvm = "-Xmx3072m", reduce_jvm = "-Xmx4096m",
  map_memory = 5120, reduce_memory = 5120,
  reduce_input_buffer_percent=0.4, reduce_parallelcopies=10,
  reduce_merge_inmem=0, task_io_sort_factor=100,
  spill_percent=0.9, reduce_shuffle_input_buffer_percent = 0.8,
  reduce_shuffle_merge_percent = 0.4,
  reduce_buffer_read = 100, map_buffer_read = 100,
  reduce_buffer_size = 10000, map_buffer_size = 100
)
swaptoLoc(FileInput, FileOutput, cluster_control=ccontrol)
```

swaptoSubser *Swap to division by subseries*

Description

Switch input key-value pairs which is division by location to the key-value pairs which is division by subseries.

Usage

```
swaptoSubser(input, output, cluster_control, model_control)
```

Arguments

`input` The path of input file on HDFS. It should be by location.

`output` The path of output file on HDFS. It is by subseries division.

`cluster_control` Should be a list object generated from `mapreduce.control` function. The list including all necessary Rhipe parameters and also user tunable MapReduce parameters.

`model_control` Should be a list object generated from `spacetime.control` function. The list including all necessary smoothing parameters of nonparametric fitting.

Details

`swaptoSubser` is used for switch division by location to division by subseries. The smoothing procedure can be applied to each subseries in parallel. So the output files will be intput to parallel seasonal fitting of stlplus.

Author(s)

Xiaosu Tong

See Also

spacetime.control, mapreduce.control

Examples

```
    FileInput <- "/wsc/tongx/spatem/tmax/sims/bystat"
    FileOutput <- "/wsc/tongx/spatem/tmax/sims/bysubser"
    ccontrol <- mapreduce.control(libLoc=.libPaths(), reduceTask=179)
    mcontrol <- spacetime.control(
      vari = "resp", time = "date", seaname = "month",
      n = 786432, n.p = 12, s.window = "periodic", t.window = 241,
      degree = 2, span = 0.015, Edeg = 2, statbytime = 2
    )
   swaptoSubser(FileInput, FileOutput, cluster_control=ccontrol, model_control=mcontrol)
```

---

swaptoTime *Swap to division by time*

Description

Switch input key-value pairs which is division by location to the key-value pairs which is division by time.

Usage

```
swaptoTime(input, output, cluster_control = mapreduce.control(),
  model_control = spacetime.control())
```

Arguments

`input` The path of input file on HDFS. It should be by location division.

`output` The path of output file on HDFS. It is by time division.

`cluster_control` Should be a list object generated from `mapreduce.control` function. The list including all necessary Rhipe parameters and also user tunable MapReduce parameters.

`model_control` Should be a list object generated from `spacetime.control` function. The list including all necessary smoothing parameters of nonparametric fitting.

Details

swaptoTime is used for switching division by location to division by time. The input key is location index, and input value is a vectorized matrix with Mlcontrol$n rows and 3 columns in order of resp, seasonal, trend. For each row of matrix, a new key-value pair is generated. Since the matrix is vectorized by column, the trend in ith row is i+Mlcontrol$n. Index j controls the index of multiple location in one time point.

Author(s)

Xiaosu Tong

See Also

spacetime.control, mapreduce.control

Examples

```
FileInput <- "/wsc/tongx/spatem/tmax/sims/bystatfit"
FileOutput <- "/wsc/tongx/spatem/tmax/sims/bymth"
ccontrol <- mapreduce.control(
  libLoc=lib.loc, reduceTask=358, io_sort=1024, BLK=128, slow_starts = 0.5,
  map_jvm = "-Xmx3072m", reduce_jvm = "-Xmx4096m",
  map_memory = 5120, reduce_memory = 5120,
  reduce_input_buffer_percent=0.2, reduce_parallelcopies=10,
  reduce_merge_inmem=0, task_io_sort_factor=100,
  spill_percent=0.9, reduce_shuffle_input_buffer_percent = 0.7,
  reduce_shuffle_merge_percent = 0.5,
  reduce_buffer_read = 100, map_buffer_read = 100,
  reduce_buffer_size = 10000, map_buffer_size = 10
)
mcontrol <- spacetime.control(
  vari = "resp", time = "date", seaname = "month",
  n = 786432, n.p = 12, s.window = "periodic", t.window = 241,
  degree = 2, span = 0.015, Edeg = 2, statbytime = 2
)
swaptoTime(FileInput, FileOutput, cluster_control=ccontrol, model_control=mcontrol)
```

Index