Package ‘glpkAPI’

September 14, 2018

Type Package
Title R Interface to C API of GLPK
Version 1.3.1
Date 2018-09-13
Depends R (>= 2.6.0)
Imports methods
Description R Interface to C API of GLPK, depends on GLPK Version >= 4.42.
SystemRequirements GLPK (>= 4.42)
License GPL-3
LazyLoad yes
NeedsCompilation yes
Repository CRAN
Date/Publication 2018-09-14 14:20:03 UTC
Author Mayo Roettger [cre],
Gabriel Gelius-Dietrich [aut],
Louis Luangkesorn [ctb]
Maintainer Mayo Roettger <mayo.roettger@hhu.de>

R topics documented:

  glpkAPI-package .................................................. 5
  addColsGLPK ..................................................... 6
  addRowsGLPK ..................................................... 7
  advBasisGLPK ................................................... 8
  bfExistsGLPK .................................................... 9
  bfUpdatedGLPK .................................................. 10
  checkDupGLPK ................................................... 11
  copyProbGLPK ................................................... 12
  cpxBasisGLPK ................................................... 13
  createIndexGLPK ............................................... 13
<table>
<thead>
<tr>
<th>R topics documented:</th>
</tr>
</thead>
<tbody>
<tr>
<td>delColsGLPK</td>
</tr>
<tr>
<td>delRowsGLPK</td>
</tr>
<tr>
<td>findColGLPK</td>
</tr>
<tr>
<td>getBfcpGLPK</td>
</tr>
<tr>
<td>getColDualIptGLPK</td>
</tr>
<tr>
<td>getColsDualIptGLPK</td>
</tr>
<tr>
<td>getColsPrimGLPK</td>
</tr>
<tr>
<td>getColsUppBndsGLPK</td>
</tr>
<tr>
<td>getInteriorParmGLPK</td>
</tr>
<tr>
<td>getMIPParmGLPK</td>
</tr>
<tr>
<td>getNumIntGLPK</td>
</tr>
<tr>
<td>getObjCoefsGLPK</td>
</tr>
<tr>
<td>getObjValGLPK</td>
</tr>
<tr>
<td>getPrimStatGLPK</td>
</tr>
</tbody>
</table>
R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>getRiiGLPK</td>
<td>56</td>
</tr>
<tr>
<td>getRowDualGLPK</td>
<td>57</td>
</tr>
<tr>
<td>getRowDualIptGLPK</td>
<td>58</td>
</tr>
<tr>
<td>getRowLowBndGLPK</td>
<td>59</td>
</tr>
<tr>
<td>getRowNameGLPK</td>
<td>60</td>
</tr>
<tr>
<td>getRowPrimGLPK</td>
<td>61</td>
</tr>
<tr>
<td>getRowPrimIptGLPK</td>
<td>62</td>
</tr>
<tr>
<td>getRowsDualGLPK</td>
<td>63</td>
</tr>
<tr>
<td>getRowsDualIptGLPK</td>
<td>63</td>
</tr>
<tr>
<td>getRowsLowBndsGLPK</td>
<td>64</td>
</tr>
<tr>
<td>getRowsPrimGLPK</td>
<td>65</td>
</tr>
<tr>
<td>getRowsPrimIptGLPK</td>
<td>66</td>
</tr>
<tr>
<td>getRowsStatGLPK</td>
<td>66</td>
</tr>
<tr>
<td>getRowsStatIptGLPK</td>
<td>67</td>
</tr>
<tr>
<td>getRowsTypesGLPK</td>
<td>68</td>
</tr>
<tr>
<td>getRowsUppBndsGLPK</td>
<td>69</td>
</tr>
<tr>
<td>getRowTypeGLPK</td>
<td>70</td>
</tr>
<tr>
<td>getRowUppBndGLPK</td>
<td>71</td>
</tr>
<tr>
<td>getSimplexParmGLPK</td>
<td>72</td>
</tr>
<tr>
<td>getSjGLPK</td>
<td>73</td>
</tr>
<tr>
<td>getSolStatGLPK</td>
<td>74</td>
</tr>
<tr>
<td>getSolStatIptGLPK</td>
<td>75</td>
</tr>
<tr>
<td>getUnbndRayGLPK</td>
<td>76</td>
</tr>
<tr>
<td>glpkConstants</td>
<td>76</td>
</tr>
<tr>
<td>glpkPtr-class</td>
<td>83</td>
</tr>
<tr>
<td>initProbGLPK</td>
<td>84</td>
</tr>
<tr>
<td>loadMatrixGLPK</td>
<td>85</td>
</tr>
<tr>
<td>mipColsValGLPK</td>
<td>86</td>
</tr>
<tr>
<td>mipColValGLPK</td>
<td>86</td>
</tr>
<tr>
<td>mipObjValGLPK</td>
<td>87</td>
</tr>
<tr>
<td>mipRowsValGLPK</td>
<td>88</td>
</tr>
<tr>
<td>mipRowValGLPK</td>
<td>89</td>
</tr>
<tr>
<td>mipStatusGLPK</td>
<td>90</td>
</tr>
<tr>
<td>mplAllocWkspGLPK</td>
<td>90</td>
</tr>
<tr>
<td>mplBuildProbGLPK</td>
<td>91</td>
</tr>
<tr>
<td>mplFreeWkspGLPK</td>
<td>92</td>
</tr>
<tr>
<td>mplGenerateGLPK</td>
<td>93</td>
</tr>
<tr>
<td>mplPostsolveGLPK</td>
<td>94</td>
</tr>
<tr>
<td>mplReadDataGLPK</td>
<td>95</td>
</tr>
<tr>
<td>mplReadModelGLPK</td>
<td>96</td>
</tr>
<tr>
<td>printIptGLPK</td>
<td>97</td>
</tr>
<tr>
<td>printMIPGLPK</td>
<td>98</td>
</tr>
<tr>
<td>printRangesGLPK</td>
<td>99</td>
</tr>
<tr>
<td>printSolGLPK</td>
<td>100</td>
</tr>
<tr>
<td>readIptGLPK</td>
<td>101</td>
</tr>
<tr>
<td>readLPGLPK</td>
<td>102</td>
</tr>
<tr>
<td>readMIPGLPK</td>
<td>103</td>
</tr>
<tr>
<td>readMPSGLPK</td>
<td>104</td>
</tr>
</tbody>
</table>
R topics documented:

readProbGLPK ........................................ 105
readSolGLPK ........................................... 106
return_codeGLPK ...................................... 107
scaleProbGLPK .......................................... 107
setBfcpGLPK ........................................... 108
setColBndGLPK .......................................... 109
setColKindGLPK .......................................... 110
setColNameGLPK .......................................... 111
setColsBndsGLPK ......................................... 112
setColsBndsObjCoefsGLPK .............................. 113
setColsKindGLPK .......................................... 114
setColsNamesGLPK ........................................ 115
setColStatGLPK ........................................... 116
setDefaultIptParmGLPK .............................. 117
setDefaultMIPParmGLPK .............................. 117
setDefaultSmpParmGLPK .............................. 118
setInteriorParmGLPK .................................... 119
setMatColGLPK ........................................... 120
setMatRowGLPK ........................................... 121
setMIPParmGLPK ........................................... 122
setObjCoefGLPK ........................................... 123
setObjCoefsGLPK .......................................... 124
setObjDirGLPK ........................................... 125
setObjNameGLPK ........................................... 126
setProbNameGLPK .......................................... 127
setRhsZeroGLPK ........................................... 128
setRiiGLPK ............................................. 128
setRowBndGLPK ........................................... 129
setRowNameGLPK .......................................... 130
setRowsBndsGLPK ......................................... 131
setRowsNamesGLPK ........................................ 132
setRowStatGLPK ........................................... 133
setSimplexParmGLPK ..................................... 134
setSjjGLPK ............................................. 135
solveInteriorGLPK ....................................... 136
solveMIPGLPK ............................................ 137
solveSimplexExactGLPK ............................... 138
solveSimplexGLPK ......................................... 139
sortMatrixGLPK .......................................... 140
status_codeGLPK ....................................... 140
stdBasisGLPK .......................................... 141
termOutGLPK ............................................ 142
unscaleProbGLPK ......................................... 143
versionGLPK ............................................. 143
warmUpGLPK ............................................. 144
writeIptGLPK ............................................ 145
writeLPGLPK ............................................. 146
writeMIPGLPK ............................................ 147
Description

A low level interface to the GNU Linear Programming Kit (GLPK).

Details

The package glpkAPI provides access to the callable library of the GNU Linear Programming Kit from within R.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldor.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

Examples

```r
# load package
library(glpkAPI)

# preparing the model
lp <- initProbGLPK()

# model data
nrows <- 5
cols <- 8

# constraint matrix
ne <- 14
ia <- c(1, 5, 1, 2, 2, 3, 1, 4, 1, 5, 3, 4, 1, 5)
ja <- c(1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 8, 8)
ar <- c(3.0, 5.6, 1.0, 2.0, 1.1, 1.0, -2.0, 2.8, -1.0, 1.0, 1.0, -1.2, -1.0, 1.9)

# objective function
obj <- c(1, 0, 0, 0, 2, 0, 0, -1)
```
# upper and lower bounds of the rows
rlower <- c(2.5, -1000, 4, 1.8, 3)
rupper <- c(1000, 2.1, 4, 5, 15)

# upper and lower bounds of the columns
clower <- c(2.5, 0, 0, 0.5, 0, 0)
cupper <- c(1000, 4.1, 1, 1, 4, 1000, 1000, 1000, 4.3)

direction of optimization
setObjDirGLPK(lp, GLP_MIN)

# add rows and columns
addRowsGLPK(lp, nrows)
addColsGLPK(lp, ncols)

setColsBndsObjCoefsGLPK(lp, c(1:ncols), clower, cupper, obj)
setRowsBndsGLPK(lp, c(1:nrows), rlower, rupper)

# load constraint matrix
loadMatrixGLPK(lp, ne, ia, ja, ar)

# solve lp problem
solveSimplexGLPK(lp)

# retrieve the results
getSolStatGLPK(lp)
getObjValGLPK(lp)
getColsPrimGLPK(lp)

# remove problem object
delProbGLPK(lp)

---

addColsGLPK  Add Columns to a GLPK Problem Object

Description

Low level interface function to the GLPK function glp_add_cols. Consult the GLPK documentation for more detailed information.

Usage

addColsGLPK(lp, ncols)

Arguments

lp          An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
ncols   The number of columns to add.
addRowsGLPK

Details
Interface to the C function addCols which calls the GLPK function glp_add_cols.

Value
The ordinal number of the first new column added to the problem object is returned.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee
The GNU GLPK home page at http://www.gnu.org/software/glpk/glpk.html

addRowsGLPK  Add Rows to a GLPK Problem Object

Description
Low level interface function to the GLPK function glp_add_rows. Consult the GLPK documentation for more detailed information.

Usage
addRowsGLPK(lp, nrows)

Arguments
lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

nrows  The number of rows to add.

Details
Interface to the C function addRows which calls the GLPK function glp_add_rows.

Value
The ordinal number of the first new row added to the problem object is returned.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
References

Based on the package **glpk** by Lopaka Lee

---

**advBasisGLPK**  
*Contract Advanced Initial LP Basis*

Description

Low level interface function to the GLPK function `glp_adv_basis`. Consult the GLPK documentation for more detailed information.

Usage

```r
advBasisGLPK(lp)
```

Arguments

- `lp`  
  An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function `advBasis` which calls the GLPK function `glp_adv_basis`.

Value

NULL

Author(s)

- Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
- Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package **glpk** by Lopaka Lee
Check if the basis factorization exists

**Description**

Low level interface function to the GLPK function `glp_bf_exists`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
bfExistsGLPK(lp)
```

**Arguments**

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**

Interface to the C function `bfExists` which calls the GLPK function `glp_bf_exists`.

**Value**

Returns non-zero if the basis factorization for the specified problem object exists. Otherwise the routine returns zero.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

bfUpdatedGLPK  

*Check if the basis factorization has been updated*

**Description**

Low level interface function to the GLPK function `glp_bf_updated`. Consult the GLPK documentation for more detailed information.

**Usage**

```
bfUpdatedGLPK(lp)
```

**Arguments**

- `lp`: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**

Interface to the C function `bfUpdated` which calls the GLPK function `glp_bf_updated`.

**Value**

Returns non-zero if the basis factorization has been updated at least once. Otherwise the routine returns zero.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

checkDupGLPK

Check for Duplicate Elements in Sparse Matrix

Description

Low level interface function to the GLPK function glp_check_dup. Consult the GLPK documentation for more detailed information.

Usage

checkDupGLPK(m, n, ne, ia, ja)

Arguments

m Number of rows in the matrix.

n Number of columns in the matrix.

ne Number of non-zero elements in the matrix.

ia Row indices of the non-zero elements.

ja Column indices of the non-zero elements.

Details

Interface to the C function checkDup which calls the GLPK function glp_check_dup.

Value

Returns one of the following values:

∅ No duplikate elements.

-ν Indices ia[k] or ja[k] are out of range.

+ν Element (ia[k], ja[k]) is duplicate.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee

The GNU GLPK home page at http://www.gnu.org/software/glpk/glpk.html
copyProbGLPK  

Copy problem object content

Description

Low level interface function to the GLPK function glp_copy_prob. Consult the GLPK documentation for more detailed information.

Usage

copyProbGLPK(lp, clp, name = GLP_OFF)

Arguments

lp  
An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

clp  
A pointer to a GLPK problem object (destination).

name  
If set to GLP_ON, the routine copies all symbolic names; otherwise (GLP_OFF) not.

Details

Interface to the C function copyProb which calls the GLPK function glp_copy_prob.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.


See Also

glpkConstants, section ‘enable/disable flag’.
cpxBasisGLPK

Construct Bixby's initial LP basis

Description
Low level interface function to the GLPK function glp_cpx_basis. Consult the GLPK documentation for more detailed information.

Usage
cpxBasisGLPK(lp)

Arguments
lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function cpxBasis which calls the GLPK function glp_cpx_basis.

Value
NULL

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

createIndexGLPK

Create the Name Index

Description
Low level interface function to the GLPK function glp_create_index. Consult the GLPK documentation for more detailed information.

Usage
createIndexGLPK(lp)
Arguments

lp  An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function createIndex which calls the GLPK function glp_create_index.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

---

`delColsGLPK`  *Delete Columns from Problem Object*

Description

Low level interface function to the GLPK function glp_del_cols. Consult the GLPK documentation for more detailed information.

Usage

```
delColsGLPK(lp, ncols, j)
```

Arguments

lp  An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
ncols  Number of columns to delete.
j  Ordinal numbers of columns to delete.

Details

Interface to the C function delCols which calls the GLPK function glp_del_cols.
deleteIndexGLPK

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

Description

Low level interface function to the GLPK function glp_delete_index. Consult the GLPK documentation for more detailed information.

Usage

deletemIndexGLPK(lp)

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function deleteIndex which calls the GLPK function glp_delete_index.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
**delProbGLPK**  
*Delete Problem Object*

**Description**  
Low level interface function to the GLPK function `glp_delete_prob`. Consult the GLPK documentation for more detailed information.

**Usage**  
```r
delProbGLPK(lp)
```

**Arguments**  
- `lp`  
  An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**  
Interface to the C function `delProb` which calls the GLPK function `glp_delete_prob`.

**Value**  
NULL

**Author(s)**  
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>  
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**  
Based on the package `glpk` by Lopaka Lee.  

---

**delRowsGLPK**  
*Delete Rows from Problem Object*

**Description**  
Low level interface function to the GLPK function `glp_del_rows`. Consult the GLPK documentation for more detailed information.

**Usage**  
```r
delRowsGLPK(lp, nrows, i)
```
Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
nrows Number of rows to delete.
i Ordinal numbers of rows to delete.

Details

Interface to the C function delRows which calls the GLPK function glp_del_rows.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

---

eraseProbGLPK Erase problem object content

Description

Low level interface function to the GLPK function glp_erase_prob. Consult the GLPK documentation for more detailed information.

Usage

eraseProbGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function eraseProb which calls the GLPK function glp_erase_prob.
factorizeGLPK

Value
NULL

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

factorizeGLPK  Compute the basis factorization

Description
Low level interface function to the GLPK function glp_factorize. Consult the GLPK documentation for more detailed information.

Usage
factorizeGLPK(lp)

Arguments
lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function factorize which calls the GLPK function glp_factorize.

Value
Returns zero if the basis factorization has been successfully computed. Otherwise the routine returns non-zero.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
findColGLPK

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘return codes’.

findColGLPK Find Column by its Name

Description

Low level interface function to the GLPK function glp_find_col. Consult the GLPK documentation for more detailed information.

Usage

findColGLPK(lp, cname)

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
cname A column name.

Details

Interface to the C function findCol which calls the GLPK function glp_find_column.

Value

Returns the ordinal number of a column, which is assigned the specified cname.

Note

Before calling findColGLPK for the first time on a problem object lp, an index has to created via a call to createIndexGLPK.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
findRowGLPK  

Find Row by its Name

Description

Low level interface function to the GLPK function glp_find_row. Consult the GLPK documentation for more detailed information.

Usage

findRowGLPK(lp, rname)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lp</td>
<td>An object of class &quot;glpPtr&quot; as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.</td>
</tr>
<tr>
<td>rname</td>
<td>A row name.</td>
</tr>
</tbody>
</table>

Details

Interface to the C function findRow which calls the GLPK function glp_find_row.

Value

Returns the ordinal number of a row, which is assigned the specified rname.

Note

Before calling findRowGLPK for the first time on a problem object lp, an index has to created via a call to createIndexGLPK.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getBfcpGLPK

Retrieve Basis Factorization Control parameters

Description

Returns the names and values of members in the structure glp_bfcp. Consult the GLPK documentation for more detailed information.

Usage

getBfcpGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getBfcp.

Value

The function returns a list.

integer The names and corresponding values of all integer control parameters in glp_bfcp.
double The names and corresponding values of all double control parameters in glp_bfcp.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘Control Parameters’.
getBheadGLPK

Retrieves Basis Header Information

Description

Low level interface function to the GLPK function glp_get_bhead. Consult the GLPK documentation for more detailed information.

Usage

getBheadGLPK(lp, k)

Arguments

lp
An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

k
Index of the basic variable.

Details

Interface to the C function getBhead which calls the GLPK function glp_get_bhead.

Value

Index of the auxiliary/structural variable.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
Description

Low level interface function to the GLPK function `glp_get_col_bind`. Consult the GLPK documentation for more detailed information.

Usage

getCbindGLPK(lp, j)

Arguments

`lp` An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

`j` Structural variable j.

Details

Interface to the C function `getCbind` which calls the GLPK function `glp_get_col_bind`.

Value

Index of the basic variable.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.
**getColDualGLPK**

*Retrieve Column Dual Value*

**Description**

Low level interface function to the GLPK function `glp_get_col_dual`. Consult the GLPK documentation for more detailed information.

**Usage**

`getColDualGLPK(lp, j)`

**Arguments**

- `lp`: An object of class `"glpkPtr"` as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `j`: Column number j.

**Details**

Interface to the C function `getColdual` which calls the GLPK function `glp_get_col_dual`.

**Value**

Column dual value

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

getColDualIptGLPK

Get Column Dual Value

Description

Low level interface function to the GLPK function glp_ipt_col_dual. Consult the GLPK documentation for more detailed information.

Usage

getColDualIptGLPK(lp, j)

Arguments

lp An object of class \texttt{glpPtr} as returned by \texttt{initProbGLPK}. This is basically a pointer to a GLPK problem object.

j Column number \(j\).

Details

Interface to the C function \texttt{getColDualIpt} which calls the GLPK function \texttt{glp_ipt_col_dual}.

Value

Column dual value

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package \texttt{glpk} by Lopaka Lee.

The GNU GLPK home page at \url{http://www.gnu.org/software/glpk/glpk.html}. 
getColKindGLPK

Retrieve Column Kind

Description

Low level interface function to the GLPK function glp_get_col_kind. Consult the GLPK documentation for more detailed information.

Usage

getColKindGLPK(lp, j)

Arguments

lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

j  Column number j.

Details

Interface to the C function getColKind which calls the GLPK function glp_get_col_kind.

Value

Column Kind

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
**getColLowBndGLPK**

*Retrieve Column Lower Bound*

---

**Description**

Low level interface function to the GLPK function `glp_get_col_lb`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
getColLowBndGLPK(lp, j)
```

**Arguments**

- **lp**: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **j**: Column number `j`.

**Details**

Interface to the C function `getColLowBnd` which calls the GLPK function `glp_get_col_lb`.

**Value**

The lower bound of the `j`-th column (the corresponding structural variable) is returned.

**Author(s)**

Gabriel Gelius-Dietrich <geliidie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

getColNameGLPK

Retrieve Column Name

Description

Low level interface function to the GLPK function `glp_get_col_name`. Consult the GLPK documentation for more detailed information.

Usage

```
getColNameGLPK(lp, j)
```

Arguments

- **lp**: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **j**: Column number `j`.

Details

Interface to the C function `getColName` which calls the GLPK function `glp_get_col_name`.

Value

The assigned name of the `j`-th column is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.
getColPrimGLPK

Retrieve Column Primal Value

Description

A low level interface function to the GLPK function `glp_get_col_prim`. Consult the GLPK documentation for more detailed information.

Usage

```r
getColPrimGLPK(lp, j)
```

Arguments

- `lp`: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `j`: Column number `j`.

Details

Interface to the C function `getColPrim` which calls the GLPK function `glp_get_col_prim`.

Value

The primal value of the `j`-th column (the corresponding structural variable) is returned.

Author(s)

- Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
- Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.

getColPrimIptGLPK

Retrieve Column Primal Value

Description

Low level interface function to the GLPK function glp_ipt_col_prim. Consult the GLPK documentation for more detailed information.

Usage

getColPrimIptGLPK(lp, j)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
j Column number j.

Details

Interface to the C function getColPrimIpt which calls the GLPK function glp_ipt_col_prim.

Value

The primal value of the j-th column (the corresponding structural variable) is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
getColsDualGLPK

Retrieve Column Dual Value of all Columns

Description
This is an advanced version of \texttt{getColDualGLPK}.

Usage
\texttt{getColsDualGLPK}(lp)

Arguments
lp An object of class "\texttt{glpkPtr}" as returned by \texttt{initProbGLPK}. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function \texttt{getColsDual} which calls the GLPK function \texttt{glp_get_col_dual}.

Value
The column dual values of all columns (structural variables) are returned.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package \texttt{glpk} by Lopaka Lee.
The GNU GLPK home page at \url{http://www.gnu.org/software/glpk/glpk.html}.

getColsDualIptGLPK

Retrieve Column Dual Value of all Columns

Description
This is an advanced version of \texttt{getColDualIptGLPK}.

Usage
\texttt{getColsDualIptGLPK}(lp)
getColsKindGLPK

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getColDualIpt which calls the GLPK function glp_ipt_col_dual.

Value

The column dual values of all columns are returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getcoldual<bd> getColDualGLPK

Retrieve Column Dual

Description

This is an advanced version of getColKindGLPK.

Usage

getcoldualLP lp LP jI

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

j Vector of column numbers.

Details

Interface to the C function getColDual which calls the GLPK function glp_ipt_col_dual.

Value

The column dual values of all columns (j) are returned.

getcoldualkind<bd> getColKindGLPK

Retrieve Column Kind

Description

This is an advanced version of getColKindGLPK.

Usage

getcoldualkindLP lp LP jI

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

j Vector of column numbers.

Details

Interface to the C function getColKind which calls the GLPK function glp_get_col_ub.

Value

The column kinds of all specified columns (j) are returned.
Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

getColsLowBndsGLPK
Retrieve Lower Bounds of Specified Columns

Description
This is an advanced version of getColLowBndGLPK. Here, j can be an integer vector.

Usage
getColsLowBndsGLPK(lp, j)

Arguments
lp
An object of class "glpkPtr" as returned by initProbGLPK. This is basically a
pointer to a GLPK problem object.

j
Vector of column numbers.

Details
Interface to the C function getColsLowBnds which calls the GLPK function glp_get_col_lb.

Value
The lower bounds of all specified columns (j) (the corresponding structural variables) are returned.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.
getColsPrimGLPK  
Retrieves all Column Primal Values

Description
This is an advanced version of getColPrimGLPK.

Usage
getColsPrimGLPK(lp)

Arguments
lp  
An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function getColsPrim which calls the GLPK functions glp_get_col_prim and glp_get_num_cols.

Value
Returns all values of the structural variables as a numeric vector.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorff.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.
getColsStatGLPK

Arguments
lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function getColsPrimIpt which calls the GLPK functions glp_ipt_col_prim and glp_get_num_cols.

Value
Returns all values of the stuctural variables as a numeric vector.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

getColsStatGLPK Retrieve Column Status of all Columns

Description
This is an advanced version of getColStatGLPK.

Usage
goingColsStatGLPK(lp)

Arguments
lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function getColsStat which calls the GLPK function glp_get_col_stat.

Value
The column status of all columns are returned.
getColStatGLPK

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package \texttt{glpk} by Lopaka Lee.
The GNU GLPK home page at \url{http://www.gnu.org/software/glpk/glpk.html}.

\begin{Verbatim}
getColStatGLPK
\end{Verbatim}

Retrieve Column Status

Description
Low level interface function to the GLPK function \texttt{glp_get_col_stat}. Consult the GLPK documentation for more detailed information.

Usage
\begin{verbatim}
getColStatGLPK(lp, j)
\end{verbatim}

Arguments
\begin{verbatim}
lp An object of class \texttt{glpPtr} as returned by \texttt{initProbGLPK}. This is basically a pointer to a GLPK problem object.
j Column number j.
\end{verbatim}

Details
Interface to the C function \texttt{getColStat} which calls the GLPK function \texttt{glp_get_col_stat}.

Value
Column status

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package \texttt{glpk} by Lopaka Lee.
The GNU GLPK home page at \url{http://www.gnu.org/software/glpk/glpk.html}.

See Also
\texttt{glpkConstants}, section ‘LP/MIP problem object’.
**getColsUppBndsGLPK**

Retrieve Upper Bounds of Specified Columns

**Description**

This is an advanced version of `getColUppBndGLPK`. Here, \(j\) can be an integer vector.

**Usage**

```r
getColsUppBndsGLPK(lp, j)
```

**Arguments**

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `j` Vector of column numbers.

**Details**

Interface to the C function `getColsUppBnds` which calls the GLPK function `glp_get_col_ub`.

**Value**

The upper bounds of all specified columns (\(j\)) (the corresponding structural variable) is returned.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

getColTypeGLPK  

Retrieves Column Type

**Description**

Low level interface function to the GLPK function `glp_get_col_type`. Consult the GLPK documentation for more detailed information.

**Usage**

`getColTypeGLPK(lp, j)`

**Arguments**

- **lp**: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **j**: Column number j.

**Details**

Interface to the C function `getColType` which calls the GLPK function `glp_get_col_type`.

**Value**

The type of the j-th column (the corresponding structural variable) is returned.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

**See Also**

`glpkConstants`, section ‘LP/MIP problem object’.
getColUppBndGLPK

Retrieve Column Upper Bound

Description

Low level interface function to the GLPK function glp_get_col_ub. Consult the GLPK documentation for more detailed information.

Usage

getColUppBndGLPK(lp, j)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

j Column number j.

Details

Interface to the C function getColUppBnd which calls the GLPK function glp_get_col_ub.

Value

The upper bound of the j-th column (the corresponding structural variable) is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getDualStatGLPK  Retrieve Status of Dual Basic Solution

Description

Low level interface function to the GLPK function glp_get_dual_stat. Consult the GLPK documentation for more detailed information.

Usage

getDualStatGLPK(lp)

Arguments

lp  An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getDualStat which calls the GLPK function glp_get_dual_stat.

Value

Status of dual basic solution

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘LP/MIP problem object’.
getInteriorParmGLPK

Retrieves the Control Parameters for the Interior-point Method.

Description

Returns the names and values of members in the structure glp_iptcp. Consult the GLPK documentation for more detailed information.

Usage

getInteriorParmGLPK()

Details

Interface to the C function getInteriorParm.

Value

The function returns a list.

integer The names and corresponding values of all integer control parameters in glp_iptcp.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘Control Parameters’.
getMatColGLPK  Retrives Column j of the Constraint Matrix.

Description

Low level interface function to the GLPK function `glp_get_mat_col`. Consult the GLPK documentation for more detailed information.

Usage

`getMatColGLPK(lp, j)`

Arguments

- **lp**: An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **j**: Column number j.

Details

Interface to the C function `getMatCol` which calls the GLPK functions `glp_get_num_rows` and `glp_get_mat_col`.

Value

Returns NULL or a list containing the non zero elements of column j:

- **nnz**: number of non zero elements in column j
- **index**: row indices of the non zero elements in column j
- **value**: numerical values of the non zero elements in column j

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

- Based on the package `glpk` by Lopaka Lee.
getMatRowGLPK

Retrieves Row i of the Constraint Matrix.

Description

Low level interface function to the GLPK function glp_get_mat_row. Consult the GLPK documentation for more detailed information.

Usage

gematrowglpk(lp, i)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lp</td>
<td>An object of class &quot;glpkPtr&quot; as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.</td>
</tr>
<tr>
<td>i</td>
<td>Row number i.</td>
</tr>
</tbody>
</table>

Details

Interface to the C function getMatRow which calls the GLPK functions glp_get_num_cols and glp_get_mat_row.

Value

Returns NULL or a list containing the non zero elements of row i:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nnz</td>
<td>number of non zero elements in row i</td>
</tr>
<tr>
<td>index</td>
<td>column indices of the non zero elements in row i</td>
</tr>
<tr>
<td>value</td>
<td>numerical values of the non zero elements in row i</td>
</tr>
</tbody>
</table>

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
getMIPParmGLPK

Retrives the Control Parameters for MIP.

Description

Returns the names and values of members in the structure glp_iocp. Consult the GLPK documentation for more detailed information.

Usage

getMIPParmGLPK()

Details

Interface to the C function getMIPParm.

Value

The function returns a list.

- **integer**: The names and corresponding values of all integer control parameters in glp_iocp.
- **double**: The names and corresponding values of all double control parameters in glp_iocp.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

- glpkConstants, section ‘Control Parameters’.
getNumBinGLPK

Retrieve Number of Binary Columns

Description

Low level interface function to the GLPK function `glp_get_num_bin`. Consult the GLPK documentation for more detailed information.

Usage

`getNumBinGLPK(lp)`

Arguments

- `lp` An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function `getNumBin` which calls the GLPK function `glp_get_num_bin`.

Value

Number of binary columns.

Author(s)

- Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
- Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.

getNumColsGLPK

Retrieve Number of Columns

Description

Low level interface function to the GLPK function `glp_get_num_cols`. Consult the GLPK documentation for more detailed information.

Usage

`getNumColsGLPK(lp)`
getNumIntGLPK

Arguments

lp
An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getNumCols which calls the GLPK function glp_get_num_cols.

Value

Returns the current number of columns in the specified problem object.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getNumIntGLPK Retrieve Number of Integer Columns

Description

Low level interface function to the GLPK function glp_get_num_int. Consult the GLPK documentation for more detailed information.

Usage

getNumIntGLPK(lp)

Arguments

lp
An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getNumInt which calls the GLPK function glp_get_num_int.

Value

Number of integer columns.
**getNumNnzGLPK**

**Author(s)**
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**
Based on the package *glpk* by Lopaka Lee.

---

**getNumNnzGLPK** *Retrieve the Number of Constraint Coefficients*

**Description**
Low level interface function to the GLPK function `glp_get_num_nz`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
getNumNnzGLPK(lp)
```

**Arguments**

`lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**
Interface to the C function `getNumNnz` which calls the GLPK function `glp_get_num_nz`.

**Value**
Returns the number of non-zero elements in the constraint matrix of the specified problem object.

**Author(s)**
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**
Based on the package *glpk* by Lopaka Lee.
getNumRowsGLPK

Retrieve Number of Rows

description
Low level interface function to the GLPK function glp_get_num_rows. Consult the GLPK documentation for more detailed information.

Usage
getNumRowsGLPK(lp)

Arguments
lp
An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

details
Interface to the C function getNumRows which calls the GLPK function glp_get_num_rows.

value
Returns the current number of rows in the specified problem object.

author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

references
Based on the package glpk by Lopaka Lee.

---

getObjCoefGLPK

Retrieve Objective Coefficient or Constant Term

description
Low level interface function to the GLPK function glp_get_obj_coef. Consult the GLPK documentation for more detailed information.

Usage
getObjCoefGLPK(lp, j)
Arguments

lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

j  Column number j.

Details

Interface to the C function getObjCoef which calls the GLPK function glp_get_obj_coef.

Value

The objective coefficient at the j-th column (the corresponding structural variable) is returned. If j is 0, the constant term “shift” of the objective function is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getObjCoefsGLPK

Retrieve Objective Coefficients at Specified Columns and/or Constant Term

Description

This is an advanced version of getObjCoefGLPK. Here, j can be an integer vector.

Usage

getObjCoefsGLPK(lp, j)

Arguments

lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

j  Vector of column numbers.

Details

Interface to the C function getObjCoef which calls the GLPK function glp_get_obj_coef.
Value

The objective coefficient at all specified columns (j) (the corresponding structural variable) is returned. If j is 0, the constant term “shift” of the objective function is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

---

getObjDirGLPK

Retrieve Optimization Direction Flag

Description

Low level interface function to the GLPK function glp_get_obj_dir. Consult the GLPK documentation for more detailed information.

Usage

getObjDirGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getObjDir which calls the GLPK function glp_get_obj_dir.

Value

Returns the optimization direction flag.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
getObjNameGLPK

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘LP/MIP problem object’.

getobjnameGLPK Retrieve Objective Function Name

Description

Low level interface function to the GLPK function glp_get_obj_name. Consult the GLPK documentation for more detailed information.

Usage

getObjNameGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getObjName which calls the GLPK function glp_get_obj_name.

Value

The assigned name of the objective function is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
getObjValGLPK

Retrieve Objective Value

Description
Low level interface function to the GLPK function glp_get_obj_val. Consult the GLPK documentation for more detailed information.

Usage
getObjValGLPK(lp)

Arguments
lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function getObjVal which calls the GLPK function glp_get_obj_val.

Value
Returns the current value of the objective function.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

getObjValIptGLPK

Retrieve Objective Value

Description
Low level interface function to the GLPK function glp_ipt_obj_val. Consult the GLPK documentation for more detailed information.

Usage
getObjValIptGLPK(lp)
getPrimStatGLPK

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getobjValIpt which calls the GLPK function glp_ipt_obj_val.

Value

Returns the current value of the objective function.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getPrimStatGLPK Retrieve Status of Primal Basic Solution

Description

Low level interface function to the GLPK function glp.get_prim_stat. Consult the GLPK documentation for more detailed information.

Usage

getPrimStatGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getPrimStat which calls the GLPK function glp.get_prim_stat.

Value

Status of primal basic solution
getProbNameGLPK

Author(s)
   Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
   Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
   Based on the package glpk by Lopaka Lee.

See Also
   glpkConstants, section ‘LP/MIP problem object’.

getProbNameGLPK | Retrieve Problem Name

Description
   Low level interface function to the GLPK function glp_get_prob_name. Consult the GLPK documentation for more detailed information.

Usage
   getProbNameGLPK(lp)

Arguments
   lp | An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
   Interface to the C function getProbName which calls the GLPK function glp_get_prob_name.

Value
   The assigned name of the problem is returned.

Author(s)
   Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
   Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
   Based on the package glpk by Lopaka Lee.
Description

Low level interface function to the GLPK function `glp_get_row_bind`. Consult the GLPK documentation for more detailed information.

Usage

```r
getRbindGLPK(lp, i)
```

Arguments

- **lp**: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **i**: Auxiliary variable i.

Details

Interface to the C function `getRbind` which calls the GLPK function `glp_get_row_bind`.

Value

Index of the basic variable.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.
getRiiGLPK  Retrieve row scale factor

Description

Low level interface function to the GLPK function glp_get_rii. Consult the GLPK documentation for more detailed information.

Usage

getRiiGLPK(lp, i)

Arguments

lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

i  Row number i.

Details

Interface to the C function getRii which calls the GLPK function glp_get_rii.

Value

Returns the current scale factor $r_{ii}$ for row i of the specified problem object.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getRowDualGLPK

Retrieve Row Dual Value

Description

Low level interface function to the GLPK function `glp_get_row_dual`. Consult the GLPK documentation for more detailed information.

Usage

```
getRowDualGLPK(lp, i)
```

Arguments

- **lp**: An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **i**: Row number i.

Details

Interface to the C function `getRowDual` which calls the GLPK function `glp_get_row_dual`.

Value

Row dual value

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.
getRowDualIptGLPK  

**Retrieve Row Dual Value**

**Description**

Low level interface function to the GLPK function `glp_ipt_row_dual`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
getRowDualIptGLPK(lp, i)
```

**Arguments**

- **lp** An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **i** Row number i.

**Details**

Interface to the C function `getRowDualIpt` which calls the GLPK function `glp_ipt_row_dual`.

**Value**

Row dual value

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

getRowLowBndGLPK

Retrieve Row Lower Bound

Description

Low level interface function to the GLPK function glp_get_row_lb. Consult the GLPK documentation for more detailed information.

Usage

ggetRowLowBndGLPK(lp, i)

Arguments

lp
    An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

i
    Row number i.

Details

Interface to the C function getRowLowBnd which calls the GLPK function glp_get_row_lb.

Value

The lower bound of the i-th row (the corresponding auxiliary variable) is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getRowNameGLPK

Retrieve Row Name

Description

Low level interface function to the GLPK function glp_get_row_name. Consult the GLPK documentation for more detailed information.

Usage

getRowNameGLPK(lp, i)

Arguments

lp
An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

i
Row number i.

Details

Interface to the C function getRowName which calls the GLPK function glp_get_row_name.

Value

The assigned name of the i-th row is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

**getRowPrimGLPK**

*Retrieve Row Primal Value*

---

**Description**

Low level interface function to the GLPK function `glp_get_row_prim`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
getRowPrimGLPK(lp, i)
```

**Arguments**

- `lp` An object of class "glkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `i` Row number i.

**Details**

Interface to the C function `getRowPrim` which calls the GLPK function `glp_get_row_prim`.

**Value**

Row primal value

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

getRowPrimIptGLPK  Retrieve Row Primal Value

Description

Low level interface function to the GLPK function glp_ipt_row_prim. Consult the GLPK documentation for more detailed information.

Usage

ggetRowPrimIptGLPK(lp, i)

Arguments

lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

i  Row number i.

Details

Interface to the C function getRowPrimIpt which calls the GLPK function glp_ipt_row_prim.

Value

Row primal value

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

**getRowsDualGLPK**

*Retrieve Row Dual Values of all Rows*

**Description**

This is an advanced version of `getRowDualGLPK`.

**Usage**

```r
getRowsDualGLPK(lp)
```

**Arguments**

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**

Interface to the C function `getRowsDual` which calls the GLPK function `glp_get_row_stat`.

**Value**

The row dual values of all rows are returned.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

---

**getRowsDualIptGLPK**

*Retrieve Row Dual Value of all Rows*

**Description**

This is an advanced version of `getRowDualIptGLPK`.

**Usage**

```r
getRowsDualIptGLPK(lp)
```
getRowsLowBndsGLPK

Arguments

lp

An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getRowsDualIpt which calls the GLPK function glp_ipt_row_dual.

Value

The row dual values of all rows are returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getRowsLowBndsGLPK

Retrieve Lower Bounds of Specified Rows

Description

This is an advanced version of getRowLowBndGLPK. Here, i can be an integer vector.

Usage

getRowsLowBndsGLPK(lp, i)

Arguments

lp

An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

i

Vector of row numbers.

Details

Interface to the C function getRowsLowBnds which calls the GLPK function glp_get_row_lb.

Value

The lower bounds of all specified columns (i) (the corresponding auxiliary variables) are returned.
getRowsPrimGLPK

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

getRowsPrimGLPK Retrieve Row Primal Value of all Rows

Description
This is an advanced version of getRowPrimGLPK.

Usage
getRowPrimGLPK(lp)

Arguments
lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function getRowsPrim which calls the GLPK function glp_get_row_prim.

Value
The row primal values for all rows are returned.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.
getRowsPrimIptGLPK  Retrieve Row Primal Value of all Rows

Description
This is an advanced version of getRowPrimIptGLPK.

Usage
getRowsPrimIptGLPK(lp)

Arguments
lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function getRowsPrimIpt which calls the GLPK function glp_ipt_row_prim.

Value
The row primal values of all rows are returned.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

getRowsStatGLPK  Retrieve Row Status of all Rows

Description
This is an advanced version of getRowStatGLPK.

Usage
getRowsStatGLPK(lp)
getRowStatGLPK

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getRowsStat which calls the GLPK function glp_get_row_stat.

Value

The row status values of all rows are returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

getRowStatGLPK Retrieve Row Status

Description

Low level interface function to the GLPK function glp_get_row_stat. Consult the GLPK documentation for more detailed information.

Usage

ggetRowStatGLPK(lp, i)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
i Row number i.

Details

Interface to the C function getRowStat which calls the GLPK function glp_get_row_stat.

Value

Row status
getRowsTypesGLPK

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

See Also
glpkConstants, section ‘LP/MIP problem object’.

---

getRowsTypesGLPK Retrieve Types of Specified Constraints (Rows)

Description
This is an advanced version of getRowTypeGLPK. Here, i can be an integer vector.

Usage
getRowsTypesGLPK(lp, i)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a
     pointer to a GLPK problem object.

i Vector of row numbers.

Details
Interface to the C function getRowsTypes which calls the GLPK function glp_get_row_type.

Value
A numeric vector of the same length as i giving the constraint type of the specified rows.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.
See Also

  glpkConstants, section ‘type of auxiliary/structural variable’.

getRowsUppBndsGLPK
Retrieve Upper Bounds of Specified Rows

Description

This is an advanced version of getRowUppBndGLPK. Here, i can be an integer vector.

Usage

getRowsUppBndsGLPK(lp, i)

Arguments

  lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
  i Vector of row numbers.

Details

Interface to the C function getRowsUppBnds which calls the GLPK function glp_get_row_ub.

Value

The upper bounds of all specified columns (i) (the corresponding auxiliary variables) are returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
getRowTypeGLPK

Retrieve Row Type

Description

Low level interface function to the GLPK function glp_get_row_type. Consult the GLPK documentation for more detailed information.

Usage

gGettyRowTypeGLPK(lp, i)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
i Row number i.

Details

Interface to the C function getRowType which calls the GLPK function glp_get_row_type.

Value

The type of the i-th row (the corresponding auxiliary variable) is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘LP/MIP problem object’.
getRowUppBndGLPK  Retrieve Row Upper Bound

Description

Low level interface function to the GLPK function glp_get_row_ub. Consult the GLPK documentation for more detailed information.

Usage

getRowUppBndGLPK(lp, i)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
i Row number i.

Details

Interface to the C function getRowUppBnd which calls the GLPK function glp_get_row_ub.

Value

The upper bound of the i-th row (the corresponding auxiliary variable) is returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
Retrieves the Control Parameters for the Simplex Method.

Description

Returns the names and values of members in the structure glp_smcp. Consult the GLPK documentation for more detailed information.

Usage

getsimplexparmGLPK()

Details

Interface to the C function getsimplex_parm.

Value

The function returns a list.

integer The names and corresponding values of all integer control parameters in glp_smcp.
double The names and corresponding values of all double control parameters in glp_smcp.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘Control Parameters’.
getSjjGLPK

Retrieve column scale factor

Description

Low level interface function to the GLPK function glp_get_sjj. Consult the GLPK documentation for more detailed information.

Usage

getSjjGLPK(lp, j)

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
j Column number j.

Details

Interface to the C function getSjj which calls the GLPK function glp_get_sjj.

Value

Returns the current scale factor $s_{jj}$ for column j of the specified problem object.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
getSolStatGLPK  
*Determine Generic Status of the Basic Solution*

**Description**

Low level interface function to the GLPK function `glp_get_status`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
getsolstatglpk(lp)
```

**Arguments**

- **lp**: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**

Interface to the C function `getsolstat` which calls the GLPK function `glp_get_status`.

**Value**

Returns the generic status of the current basic solution for the specified problem object.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.


**See Also**

`glpkConstants`, section ‘LP/MIP problem object’.
getSolStatIptGLPK

Determine Solution Status

Description

Low level interface function to the GLPK function glp_ipt_status. Consult the GLPK documentation for more detailed information.

Usage

getSolStatIptGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function getSolStatIpt which calls the GLPK function glp_ipt_status.

Value

Returns the generic status of the current basic solution for the specified problem object.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘LP/MIP problem object’.
getUnbndRayGLPK  \textit{Determine Variable Causing Unboundedness}

\textbf{Description}

Low level interface function to the GLPK function \texttt{glp.get.unbnd-ray}. Consult the GLPK documentation for more detailed information.

\textbf{Usage}

getUnbndRayGLPK(lp)

\textbf{Arguments}

\begin{itemize}
\item \texttt{lp} An object of class "\texttt{glpkPtr}" as returned by \texttt{initProbGLPK}. This is basically a pointer to a GLPK problem object.
\end{itemize}

\textbf{Details}

Interface to the C function getUnbndRay which calls the GLPK function \texttt{glp.get.unbnd-ray}.

\textbf{Value}

Returns the number k of a variable, which causes primal or dual unboundedness.

\textbf{Author(s)}

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

\textbf{References}

Based on the package \texttt{glpk} by Lopaka Lee.
The GNU GLPK home page at \url{http://www.gnu.org/software/glpk/glpk.html}.

\textbf{glpkConstants  \textit{Constants, Return and Status Codes of GLPK}}

\textbf{Description}

This is a list containing constants used by GLPK. Consult the glpk manual for more information, in particular for the control parameters.

\textbf{Control Parameters}

Simplex
The exact simplex method uses only the parameters \texttt{IT\_LIM} and \texttt{TM\_LIM}.

\textit{Interior}

\texttt{MSG\_LEV} \gets 101 \quad \text{Message level for terminal output (default: GLP\_MSG\_ALL).}
\texttt{ORD\_ALG} \gets 301 \quad \text{Ordering algorithm used prior to Cholesky factorization (default: GLP\_ORD\_AMD).}

\textit{MIP}

\texttt{MSG\_LEV} \gets 101 \quad \text{Message level for terminal output (default: GLP\_MSG\_ALL).}
\texttt{TM\_LIM} \gets 106 \quad \text{Searching time limit, in milliseconds (default: INT\_MAX).}
\texttt{OUT\_FRQ} \gets 107 \quad \text{Output frequency, in iterations (default: 5000).}
\texttt{OUT\_DLY} \gets 108 \quad \text{Output delay, in milliseconds (default: 0).}
\texttt{PRESOLVE} \gets 109 \quad \text{LP presolver option (default: GLP\_OFF).}
\texttt{BR\_TECH} \gets 601 \quad \text{Branching technique option (default: GLP\_BR\_DTH).}
\texttt{BT\_TECH} \gets 602 \quad \text{Backtracking technique option (default: GLP\_BT\_BLB).}
\texttt{PP\_TECH} \gets 603 \quad \text{Preprocessing technique option (default: GLP\_PP\_ALL).}
\texttt{FP\_HEUR} \gets 604 \quad \text{Feasibility pump heuristic option (default: GLP\_OFF).}
\texttt{GMI\_CUTS} \gets 605 \quad \text{Gomory's mixed integer cut option (default: GLP\_OFF).}
\texttt{MIR\_CUTS} \gets 606 \quad \text{Mixed integer rounding (MIR) cut option (default: GLP\_OFF).}
\texttt{COV\_CUTS} \gets 607 \quad \text{Mixed cover cut option (default: GLP\_OFF).}
\texttt{CLQ\_CUTS} \gets 608 \quad \text{Clique cut option (default: GLP\_OFF).}
\texttt{CB\_SIZE} \gets 509 \quad \text{The number of extra (up to 256) bytes allocated for each node of the branch-and-bound tree to store application-specific data.}
\texttt{BINARYIZE} \gets 610 \quad \text{LP presolver option (default: GLP\_OFF).}
\texttt{CB\_FUNC} \gets 651 \quad \text{Use a user defined callback routine \texttt{glpkCall} which is written in the file ‘glpkCall.c’. This function can be used to control the optimization process.}
\texttt{TOL\_INT} \gets 701 \quad \text{Absolute tolerance used to check if optimal solution to the current LP relaxation is integer feasible (default: 1e-8).}
\texttt{TOL\_OBJ} \gets 702 \quad \text{Relative tolerance used to check if the objective value in optimal solution to the current LP relaxation is integer feasible (default: 1e-6).}
\texttt{MIP\_GAP} \gets 703 \quad \text{The relative gap tolerance. If the relative gap for currently known best integer feasible solution is below this value, the algorithm terminates.}
TYPE <= 401  Basis factorization type (default: GLP_BF_FT).
LU_SIZE <= 402  Initial size of the Sparse Vector Area (default: 0).
PIV_LIM <= 403  computing LU-factorization of the basis matrix (default: 4).
SUHL <= 404  computing LU-factorization of the basis matrix (default: GLP_ON).
NFS_MAX <= 405  Maximal number of additional row-like factors (default: 100).
NRS_MAX <= 406  Maximal number of additional rows and columns (default: 100).
RS_SIZE <= 407  Initial size of the Sparse Vector Area (default: 0).
PIV_TOL <= 501  Threshold pivoting (Markowitz) tolerance (default: 0.10).
EPS_TOL <= 502  Epsilon tolerance (default: 1e-15).
MAX_GRO <= 503  Maximal growth of elements of factor U (default: 1e+10).
UPD_TOL <= 504  Update tolerance (default: 1e-6).

LP/MIP problem object

optimization direction flag

GLP_MIN <= 1  minimization
GLP_MAX <= 2  maximization

kind of structural variable

GLP_CV <= 1  continuous variable
GLP_IV <= 2  integer variable
GLP_BV <= 3  binary variable

type of auxiliary/structural variable

GLP_FR <= 1  free variable
GLP_LO <= 2  variable with lower bound
GLP_UP <= 3  variable with upper bound
GLP_DB <= 4  double-bounded variable
GLP_FX <= 5  fixed variable

status of auxiliary/structural variable

GLP_BS <= 1  basic variable
GLP_NL <= 2  non-basic variable on lower bound
GLP_NU <= 3  non-basic variable on upper bound
GLP_NF <= 4  non-basic free variable
GLP_NS <= 5  non-basic fixed variable

scaling options
glpkConstants

GLP_SF_GM <= 0x01 perform geometric mean scaling
GLP_SF_EQ <= 0x10 perform equilibration scaling
GLP_SF_2N <= 0x20 round scale factors to power of two
GLP_SF_SKIP <= 0x40 skip if problem is well scaled
GLP_SF_AUTO <= 0x80 choose scaling options automatically

solution indicator

GLP_SOL <= 1 basic solution
GLP_IPT <= 2 interior-point solution
GLP_MIP <= 3 mixed integer solution

solution status

GLP_UNDEF <= 1 solution is undefined
GLP_FEAS <= 2 solution is feasible
GLP_INFEAS <= 3 solution is infeasible
GLP_NOFEAS <= 4 no feasible solution exists
GLP_OPT <= 5 solution is optimal
GLP_UNBND <= 6 solution is unbounded

basis factorization control parameters

type

GLP_BF_FT <= 0x01 LUF + Forrest-Tomlin
GLP_BF_BG <= 0x02 LUF + Schur compl. + Bartels-Golub
GLP_BF_GR <= 0x03 LUF + Schur compl. + Givens rotation
GLP_BF_LUF <= 0x00 plain LU-factorization
GLP_BF_BTF <= 0x10 block triangular LU-factorization

simplex method control parameters

msg_lev message level:

GLP_MSG_OFF <= 0 no output
GLP_MSG_ERR <= 1 warning and error messages only
GLP_MSG_ON <= 2 normal output
GLP_MSG_ALL <= 3 full output
GLP_MSG_DBG <= 4 debug output

meth simplex method option:
GLP_PRIMAL <- 1 use primal simplex  
GLP_DUALP <- 2 use dual; if it fails, use primal  
GLP_DUAL <- 3 use dual simplex  

pricing pricing technique:  
GLP_PT_STD <- 0x11 standard (Dantzig rule)  
GLP_PT_PSE <- 0x22 projected steepest edge  

r_test ratio test technique:  
GLP_RT_STD <- 0x11 standard (textbook)  
GLP_RT_HAR <- 0x22 two-pass Harris’ ratio test  

interior-point solver control parameters  
ord_alg ordering algorithm:  
GLP_ORD_NONE <- 0 natural (original ordering)  
GLP_ORD_QMD <- 1 quotient minimum degree (QMD)  
GLP_ORD_AMD <- 2 approx. minimum degree (AMD)  
GLP_ORD_SYMAMD <- 3 approx. minimum degree (SYMAMD)  

integer optimizer control parameters  
br.tech branching technique:  
GLP_BR_FFV <- 1 first fractional variable  
GLP_BR_LFV <- 2 last fractional variable  
GLP_BR_MFV <- 3 most fractional variable  
GLP_BR_DTH <- 4 heuristic by Driebeck and Tomlin  
GLP_BR_HPC <- 5 hybrid pseudocost  

bt.tech backtracking technique:  
GLP_BT_DFS <- 1 depth first search  
GLP_BT_BFS <- 2 breadth first search  
GLP_BT_BLB <- 3 best local bound  
GLP_BT_BPH <- 4 best projection heuristic  

pp.tech preprocessing technique:
**glpkConstants**

GLP_PP_NONE <- 0  disable preprocessing  
GLP_PP_ROOT <- 1  preprocessing only on root level  
GLP_PP_ALL <- 2  preprocessing on all levels  

**additional row attributes**

*the row origin flag*

GLP_RF_REG <- 0  regular constraint  
GLP_RF_LAZY <- 1  "lazy" constraint  
GLP_RF_CUT <- 2  cutting plane constraint  

*the row class descriptor* klass

GLP_RF_GMI <- 1  Gomory’s mixed integer cut  
GLP_RF_MIR <- 2  mixed integer rounding cut  
GLP_RF_COV <- 3  mixed cover cut  
GLP_RF_CLQ <- 4  clique cut  

**enable/disable flag**

GLP_ON <- 1  enable something  
GLP_OFF <- 0  disable something  

**reason codes**

GLP_IROWGEN <- 0x01  request for row generation  
GLP_IBINGO <- 0x02  better integer solution found  
GLP_IHEUR <- 0x03  request for heuristic solution  
GLP_ICUTGEN <- 0x04  request for cut generation  
GLP_IBRANCH <- 0x05  request for branching  
GLP_ISELECT <- 0x06  request for subproblem selection  
GLP_IPREPRO <- 0x07  request for preprocessing  

**branch selection indicator**

GLP_NO_BRNCH <- 0  select no branch  
GLP_DN_BRNCH <- 1  select down-branch
GLP_UP_BRNCH <- 2  select up-branch

return codes

GLP_EBADB <- 0x01  invalid basis
GLP_ESING <- 0x02  singular matrix
GLP_ECOND <- 0x03  ill-conditioned matrix
GLP_EBOUND <- 0x04  invalid bounds
GLP_EFAIL <- 0x05  solver failed
GLP_EOBJLL <- 0x06  objective lower limit reached
GLP_EOBJUL <- 0x07  objective upper limit reached
GLP_EITLIM <- 0x08  iteration limit exceeded
GLP_ETMLIM <- 0x09  time limit exceeded
GLP_ENOPFS <- 0x0A  no primal feasible solution
GLP_ENODFS <- 0x0B  no dual feasible solution
GLP_EROOT <- 0x0C  root LP optimum not provided
GLP_ESTOP <- 0x0D  search terminated by application
GLP_EMIPGAP <- 0x0E  relative mip gap tolerance reached
GLP_ENOFEAS <- 0x0F  no primal/dual feasible solution
GLP_ENOCVG <- 0x10  no convergence
GLP_EINSTAB <- 0x11  numerical instability
GLP_EDATA <- 0x12  invalid data
GLP_ERANGE <- 0x13  result out of range

condition indicator

GLP_KKT_PE <- 1  primal equalities
GLP_KKT_PB <- 2  primal bounds
GLP_KKT_DE <- 3  dual equalities
GLP_KKT_DB <- 4  dual bounds
GLP_KKT_CS <- 5  complementary slackness

MPS file format

GLP_MPS DECK <- 1  fixed (ancient)
GLP_MPS FILE <- 2  free (modern)
Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

See Also
status_codeGLPK, return_codeGLPK

glpkPtr-class Class "glpkPtr"

Description
Structure of the class "glpkPtr". Objects of that class are used to hold pointers to C structures used by GLPK.

Objects from the Class
Objects can be created by calls of the form
test <- initProbGLPK() or
test <- mplAllocWkspGLPK().

Slots

  glpkPtrType: Object of class "character" giving the pointer type.
  glpkPointer: Object of class "externalptr" containing the pointer to a C structure.

Methods

  isGLPKpointer signature(object = "glpkPtr"): returns TRUE if glpkPointer(object) is a pointer to a GLPK problem object, otherwise FALSE.
  isNULLpointerGLPK signature(object = "glpkPtr"): returns TRUE if glpkPointer(object) is a NULL pointer, otherwise FALSE.
  isTRWKSpointer signature(object = "glpkPtr"): returns TRUE if glpkPointer(object) is a pointer to a MathProg translator workspace, otherwise FALSE.
  glpkPointer signature(object = "glpkPtr"): gets the glpkPointer slot.
  glpkPtrType signature(object = "glpkPtr"): gets the glpkPtrType slot.
  glpkPtrType<- signature(object = "glpkPtr"): sets the glpkPtrType slot.
initProbGLPK

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

See Also
mplAllocWkspGLPK and initProbGLPK.

Examples

showClass("glpkPtr")

initProbGLPK Create a GLPK Problem Object

Description
Low level interface function to the GLPK function glp_create_prob. Consult the GLPK documentation for more detailed information.

Usage

initProbGLPK(ptrtype = "glpk_prob")

Arguments

ptrtype A name for the pointer to a GLPK problem object.

Details
Interface to the C function initProb which calls the GLPK function glp_create_prob.

Value
An instance of class "glpkPtr".

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
**loadMatrixGLPK**

**References**

Based on the package **glpk** by Lopaka Lee.

**See Also**

"glpkPtr".

---

**loadMatrixGLPK**  
*Load/Replace the Whole Constraint Matrix*

**Description**

Low level interface function to the GLPK function `glp_load_matrix`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
loadMatrixGLPK(lp, ne, ia, ja, ra)
```

**Arguments**

- **lp**
  An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **ne**
  Number of non-zero elements.
- **ia**
  Row indices of the non-zero elements.
- **ja**
  Column indices of the non-zero elements.
- **ra**
  The numeric values of the constraint coefficients.

**Details**

Interface to the C function `loadMatrix` which calls the GLPK function `glp_load_matrix`.

**Value**

`NULL`

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package **glpk** by Lopaka Lee.
mipColsValGLPK  Retrieve Column Value of all Columns

Description
This is an advanced version of mipColValGLPK.

Usage
mipColsValGLPK(lp)

Arguments
lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details
Interface to the C function mipColsVal which calls the GLPK function glp_mip_col_val.

Value
The column values of all columns are returned.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

mipColValGLPK  Retrieve Column Value

Description
Low level interface function to the GLPK function glp_mip_col_val. Consult the GLPK documentation for more detailed information.

Usage
mipColValGLPK(lp, j)
mipObjValGLPK

**Arguments**

- `lp` An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `j` Column number `j`.

**Details**

Interface to the C function `mipColVal` which calls the GLPK function `glp_mip_col_val`.

**Value**

Column value of column `j`.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

---

mipObjValGLPK  

**Retrieve Objective Value**

**Description**

Low level interface function to the GLPK function `glp_mip_obj_val`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
mipObjValGLPK(lp)
```

**Arguments**

- `lp` An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**

Interface to the C function `mipObjVal` which calls the GLPK function `glp_mip_obj_val`.

**Value**

Objective value.
Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

---

mipRowsValGLPK  Retrieve Row Value of all Rows

Description

This is an advanced version of mipRowValGLPK.

Usage

mipRowsValGLPK(lp)

Arguments

lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function mipRowsVal which calls the GLPK function glp_mip_row_val.

Value

The row values of all rows are returned.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
mipRowValGLPK

---

**mipRowValGLPK**

*Retrieve Row Value*

---

**Description**

Low level interface function to the GLPK function `glp_mip_row_val`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
mipRowValGLPK(lp, i)
```

**Arguments**

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `i` Row number i.

**Details**

Interface to the C function `mipRowVal` which calls the GLPK function `glp_mip_row_val`.

**Value**

Row value of row i.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

**mplAllocWkspGLPK**

*Allocate Translator Workspace*

**Description**

Low level interface function to the GLPK function `glp_mpl_alloc_wksp`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
mplAllocWkspGLPK(ptrtype = "tr_wksp")
```

**Arguments**

- `ptrtype` An object of class `"glpkPtr"` as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**

Interface to the C function `mipStatus` which calls the GLPK function `glp_mip_status`.

**Value**

Status of MIP Solution.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorfd.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.


---

**mipStatusGLPK**

*Determine Status of MIP Solution*

**Description**

Low level interface function to the GLPK function `glp_mip_status`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
mipStatusGLPK(lp)
```

**Arguments**

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**

Interface to the C function `mipStatus` which calls the GLPK function `glp_mip_status`.

**Value**

Status of MIP Solution.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorfd.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

mplBuildProbGLPK

Arguments

ptrtype A name for the pointer to a translator workspace.

Details

Interface to the C function mplAllocWksp which calls the GLPK function glp_mpl_alloc_wksp.

Value

An instance of class "glpkPtr".

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

"glpkPtr".

mplBuildProbGLPK	Build Problem Instance From Model

Description

Low level interface function to the GLPK function glp_mpl_build_prob. Consult the GLPK documentation for more detailed information.

Usage

cmplBuildProbGLPK(wk, lp)

Arguments

wk An object of class "glpkPtr" as returned by mplAllocWkspGLPK. This is basically a pointer to a GLPK translocator workspace.
lp A pointer to a GLPK problem object.

Details

Interface to the C function mplBuildProb which calls the GLPK function glp_mpl_build_prob.
mplFreeWkspGLPK

Value

Returns zero on success, otherwise it returns non-zero.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

mplAllocWkspGLPK, mplFreeWkspGLPK, mplGenerateGLPK, mplPostsolveGLPK, mplReadDataGLPK
and mplReadModelGLPK.

mplFreeWkspGLPK

Free Translator Workspace

Description

Low level interface function to the GLPK function glp_mpl_free_wksp. Consult the GLPK documentation for more detailed information.

Usage

mplFreeWkspGLPK(wk)

Arguments

wk

An object of class "glpkPtr" as returned by mplAllocWkspGLPK. This is basically a pointer to a GLPK translocator workspace.

Details

Interface to the C function mplFreeWksp which calls the GLPK function glp_mpl_free_wksp.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
mplGenerateGLPK

References
Based on the package glpk by Lopaka Lee.

See Also
mplAllocWKspGLPK, mplBuildProbGLPK, mplGenerateGLPK, mplPostsolveGLPK, mplReadDataGLPK
and mplReadModelGLPK.

mplGenerateGLPK Generate the Model

Description
Low level interface function to the GLPK function glp_mpl_generate. Consult the GLPK documentation for more detailed information.

Usage
mplGenerateGLPK(wk, fname = NULL)

Arguments
wk An object of class "glpkPtr" as returned by mplAllocWKspGLPK. This is basically a pointer to a GLPK translocator workspace.
fname The name of the text file to be written out.

Details
Interface to the C function mplGenerate which calls the GLPK function glp_mpl_generate.

Value
Returns zero on success, otherwise it returns non-zero.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.
See Also

mplAllocWkspGLPK, mplBuildProbGLPK, mplFreeWkspGLPK, mplPostsolveGLPK, mplReadDataGLPK
and mplReadModelGLPK.

mplPostsolveGLPK             Postsolve Model

Description

Low level interface function to the GLPK function glp_mpl_postsolve. Consult the GLPK documentation for more detailed information.

Usage

mplPostsolveGLPK(wk, lp, sol)

Arguments

wk     An object of class "glpkPtr" as returned by mplAllocWkspGLPK. This is basically a pointer to a GLPK translocator workspace.
lp     A pointer to a GLPK problem object.
sol    Type of solution to be copied to the translator workspace, for possible values, see glpkConstants, section 'LP/MIP problem object'.

Details

Interface to the C function mplPostsolve which calls the GLPK function glp_mpl_postsolve.

Value

Returns zero on success, otherwise it returns non-zero.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

mplAllocWkspGLPK, mplBuildProbGLPK, mplFreeWkspGLPK, mplGenerateGLPK, mplReadDataGLPK
and mplReadModelGLPK.
**mplReadDataGLPK**  

*Read and Translate Data Section*

### Description

Low level interface function to the GLPK function `glp_mpl_read_data`. Consult the GLPK documentation for more detailed information.

### Usage

```r
mplReadDataGLPK(wk, fname)
```

### Arguments

- **wk**
  An object of class "`glpPtr" as returned by `mplAllocWkspGLPK`. This is basically a pointer to a GLPK translocator workspace.

- **fname**
  The name of the data file to be read in.

### Details

Interface to the C function `mplReadData` which calls the GLPK function `glp_mpl_read_data`.

### Value

Returns zero on success, otherwise it returns non-zero.

### Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>  
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

### References

Based on the package `glpk` by Lopaka Lee.  

### See Also

`mplAllocWkspGLPK`, `mplBuildProbGLPK`, `mplFreeWkspGLPK`, `mplGenerateGLPK`, `mplPostsolveGLPK` and `mplReadModelGLPK`.  

mplReadModelGLPK

Read and Translate Model Section

Description

Low level interface function to the GLPK function glp_mpl_read_model. Consult the GLPK documentation for more detailed information.

Usage

mplReadModelGLPK(wk, fname, skip)

Arguments

wk An object of class "glpkPtr" as returned by mplAllocWkspGLPK. This is basically a pointer to a GLPK translocator workspace.
fname The name of the model file to be read in.
skip Flag, how to treat the data section.

Details

Interface to the C function mplReadModel which calls the GLPK function glp_mpl_read_model.

Value

Returns zero on success, otherwise it returns non-zero.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

mplAllocWkspGLPK, mplBuildProbGLPK, mplFreeWkspGLPK, mplGenerateGLPK, mplPostsolveGLPK
and mplReadDataGLPK.
printIptGLPK  Write Interior-Point Solution in Printable Format

Description

Low level interface function to the GLPK function glp_print_ipt. Consult the GLPK documentation for more detailed information.

Usage

printIptGLPK(lp, fname)

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fname The name of the text file to be written out.

Details

Interface to the C function printIpt which calls the GLPK function glp_print_ipt.

Value

Returns zero on success, otherwise it returns non-zero and prints an error message.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

printSolGLPK, readSolGLPK, writeSolGLPK, readIptGLPK, writeIptGLPK, printMIPGLPK, readMIPGLPK and writeMIPGLPK.
printMIPGLPK  Write Interior-Point Solution in Printable Format

Description

Low level interface function to the GLPK function glp_print_mip. Consult the GLPK documentation for more detailed information.

Usage

printMIPGLPK(lp, fname)

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fname The name of the text file to be written out.

Details

Interface to the C function printMIP which calls the GLPK function glp_print_mip.

Value

Returns zero on success, otherwise it returns non-zero and prints an error message.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

printSolGLPK, readSolGLPK, writeSolGLPK, printIptGLPK, readIptGLPK, writeIptGLPK, readMIPGLPK and writeMIPGLPK.
Print Sensitivity Analysis Report

Description

Low level interface function to the GLPK function glp_print_ranges. Consult the GLPK documentation for more detailed information.

Usage

printrangesGLPK(lp, numrc = 0, rowcol = NULL, fname = "sar.txt")

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
numrc Length of the row/column list (argument rowcol).
rowcol Ordinal numbers of rows and columns to be analyzed.
fname A filename.

Details

Interface to the C function printRanges which calls the GLPK function glp_print_ranges.

Value

Zero on success, otherwise non-zero.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
printSolGLPK  Write Basic Solution in Printable Format

Description

Low level interface function to the GLPK function glp_print_sol. Consult the GLPK documentation for more detailed information.

Usage

printSolGLPK(lp, fname)

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fname The name of the text file to be written out.

Details

Interface to the C function printSol which calls the GLPK function glp_print_sol.

Value

Returns zero on success, otherwise it returns non-zero and prints an error message.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

readSolGLPK, writeSolGLPK, printIptGLPK, readIptGLPK, writeIptGLPK, printMIPGLPK, readMIPGLPK and writeMIPGLPK.
**Description**

Low level interface function to the GLPK function `glp_read_ipt`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
readIptGLPK(lp, fname)
```

**Arguments**

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `fname` The name of the text file to be read in.

**Details**

Interface to the C function `readIpt` which calls the GLPK function `glp_read_ipt`.

**Value**

Returns zero on success, otherwise it returns non-zero.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.


**See Also**

`printSolGLPK`, `readSolGLPK`, `writeSolGLPK`, `printIptGLPK`, `writeIptGLPK`, `printMIPGLPK`, `readMIPGLPK` and `writeMIPGLPK`. 
Description

Low level interface function to the GLPK function glp_read_lp. Consult the GLPK documentation for more detailed information.

Usage

readLPGLPK(lp, fname)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fname The name of the text file to be read in.

Details

Interface to the C function readLP which calls the GLPK function glp_read_lp.

Value

Returns zero on success, otherwise it returns non-zero and prints an error message.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

readMPSGLPK, readProbGLPK, writeMPSGLPK, writeLPGLPK and writeProbGLPK.
readMIPGLPK  

Read MIP Solution From Text File

Description

Low level interface function to the GLPK function glp_read_mip. Consult the GLPK documentation for more detailed information.

Usage

readMIPGLPK(lp, fname)

Arguments

lp  
An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

fname  
The name of the text file to be read in.

Details

Interface to the C function readMIP which calls the GLPK function glp_read_mip.

Value

Returns zero on success, otherwise it returns non-zero.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

printSolGLPK, readSolGLPK, writeSolGLPK, printIptGLPK, readIptGLPK, writeIptGLPK, printMIPGLPK and writeMIPGLPK.
Description

Low level interface function to the GLPK function glp_read_mps. Consult the GLPK documentation for more detailed information.

Usage

readMPSGLPK(lp, fmt, fname)

Arguments

lp  An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fmt  MPS format. See glpkConstants, section ‘MPS file formats’.
fname  The name of the text file to be read in.

Details

Interface to the C function readMPS which calls the GLPK function glp_read_mps.

Value

Returns zero on success, otherwise it returns non-zero and prints an error message.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

readLPGLPK, readProbGLPK, writeMPSGLPK, writeLPGLPK, writeProbGLPK and glpkConstants.
Description

Low level interface function to the GLPK function `glp_read_prob`. Consult the GLPK documentation for more detailed information.

Usage

```r
readProbGLPK(lp, fname)
```

Arguments

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `fname` The name of the text file to be read in.

Details

Interface to the C function `readProb` which calls the GLPK function `glp_read_prob`.

Value

Returns zero on success, otherwise it returns non-zero and prints an error message.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.

See Also

`readMPSGLPK`, `readLPGLPK`, `writeMPSGLPK`, `writeLPGLPK` and `writeProbGLPK`. 

readSolGLPK

**Description**

Low level interface function to the GLPK function glp_read_sol. Consult the GLPK documentation for more detailed information.

**Usage**

```r
readSolGLPK(lp, fname)
```

**Arguments**

- `lp` An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `fname` The name of the text file to be read in.

**Details**

Interface to the C function `readSol` which calls the GLPK function `glp_read_sol`.

**Value**

Returns zero on success, otherwise it returns non-zero.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

**See Also**

`printSolGLPK`, `writeSolGLPK`, `printIptGLPK`, `readIptGLPK`, `writeIptGLPK`, `printMIPGLPK`, `readMIPGLPK` and `writeMIPGLPK`
**return_codeGLPK**

*Translates a GLPK Return Code into a Human Readable String*

**Description**

Translates a GLPK return code into a human readable string.

**Usage**

```r
code = return_codeGLPK(code)
```

**Arguments**

- `code` Return code from GLPK.

**Value**

A character string associated with the GLPK return code.

**Author(s)**

- Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
- Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

**See Also**

- `glpkConstants`, section ‘return codes’.

---

**scaleProbGLPK**

*Scale Problem Data*

**Description**

Low level interface function to the GLPK function `glp_scale_prob`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
code = scaleProbGLPK(lp, opt)
```
setBfcpGLPK

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

opt Scaling option, see glpkConstants, section ‘LP/MIP problem object’ for possible values.

Details

Interface to the C function scaleProb which calls the GLPK function glp_scale_prob.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants

setBfcpGLPK Change Basis Factorization Control Parameters

Description

Sets/Changes the values of corresponding members of in the structure glp_bfcp. Consult the GLPK documentation for more detailed information.

Usage

setBfcpGLPK(lp, parm, val)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

parm A vector containing integer values or symbolic names of the control parameters to be changed (see glpkConstants, section ‘Control Parameters’).

val A vector containing the new values for the corresponding control parameters.
Details

The Arguments `parm` and `val` must have the same length. The value `val[i]` belongs to the parameter `parm[i]`.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.

See Also

`glpkConstants`

---

**setColBndGLPK**

### Set/Change Column Bounds

**Description**

Low level interface function to the GLPK function `glp_set_col_bnds`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
setColBndGLPK(lp, j, type, lb, ub)
```

**Arguments**

- `lp`: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `j`: Column number `j`.
- `type`: Column type. For possible values, see `glpkConstants`, section ‘LP/MIP problem object’.
- `lb`: Lower bound.
- `ub`: Upper bound.
Details

Interface to the C function setColBnd which calls the GLPK function glp_set_col_bnds.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants

setColKindGLPK Set Column Kind

Description

Low level interface function to the GLPK function glp_set_col_kind. Consult the GLPK documentation for more detailed information.

Usage

setColKindGLPK(lp, j, kind)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
j Column number j.
kind Kind of column number j, for possible values see glpkConstants, section 'LP/MIP problem object'.

Details

Interface to the C function setColKind which calls the GLPK function glp_set_col_kind.
setColNameGLPK

Value
NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants

---

### setColNameGLPK

*Set/Change Column Name*

**Description**

Low level interface function to the GLPK function glp_set_col_name. Consult the GLPK documentation for more detailed information.

**Usage**

```r
setColNameGLPK(lp, j, cname = NULL)
```

**Arguments**

- **lp**
  An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **j**
  Column number j.
- **cname**
  Column name.

**Details**

Interface to the C function setColName which calls the GLPK function glp_set_col_name.

**Value**

NULL
**Author(s)**
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**
Based on the package glpk by Lopaka Lee.

---

**setColsBndsGLPK**

*Set/Change Column Bounds*

**Description**
This is an advanced version of setColBndGLPK. Here, \( j \) can be an integer vector, \( lb \) and \( ub \) can be numeric vectors.

**Usage**

\[
\text{setColsBndsGLPK}(lp, j, lb, ub, type = \text{NULL})
\]

**Arguments**
- \( lp \): An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
- \( j \): Vector of column numbers.
- \( lb \): Vector of lower bounds.
- \( ub \): Vector of upper bounds.
- \( type \): Vector of variable types (default: NULL). For possible values, see glpkConstants, section ‘LP/MIP problem object’.

**Details**
Interface to the C function setColsBnds which calls the GLPK function glp_set_col_bnds.
If \( type \) is set to \text{NULL}, the type of the variables will be estimated. If \( lb[i] \) equals \( ub[i] \), variable \( j[i] \) is fixed, otherwise double bounded.

**Value**
NULL

**Author(s)**
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants

---

**setColsBndsObjCoefsGLPK**

*Set/Change Column Bounds and Objective Coefficients and/or Constant Term*

---

**Description**

This is an combined version of setColsBndsglpk and setObjCoefsglpk. Here, j can be an integer vector, lb, ub and obj_coef can be numeric vectors.

**Usage**

```r
setColsBndsObjCoefsGLPK(lp, j, lb, ub, obj_coef, type = NULL)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lp</td>
<td>An object of class &quot;glpkPtr&quot; as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.</td>
</tr>
<tr>
<td>j</td>
<td>Vector of column numbers.</td>
</tr>
<tr>
<td>lb</td>
<td>Vector of lower bounds.</td>
</tr>
<tr>
<td>ub</td>
<td>Vector of upper bounds.</td>
</tr>
<tr>
<td>obj_coef</td>
<td>Vector of objective coefficients.</td>
</tr>
<tr>
<td>type</td>
<td>Vector of variable types (default: NULL). For possible values, see glpkConstants, section ‘LP/MIP problem object’.</td>
</tr>
</tbody>
</table>

**Details**

Interface to the C function setColsBndsObjCoefs which calls the GLPK functions glp_set_col_bnds and glp_set_obj_coef.

If type is set to NULL, the type of the variables will be estimated. If lb[i] equals ub[i], variable j[i] is fixed, otherwise double bounded.

**Value**

NULL
Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

See Also
   glpkConstants

---

setColsKindGLPK  Set Column Kind for a Set of Columns

Description
This is an advanced version of setColKindGLPK. Here, j can be an integer vector.

Usage
   setColsKindGLPK(lp, j, kind)

Arguments
   lp   An object of class "glpkPtr" as returned by initProbGLPK. This is basically a
         pointer to a GLPK problem object.
   j    An integer vector of column indices.
   kind An integer vector of column kinds, for possible values see glpkConstants, section ‘LP/MIP problem object’.

Details
Interface to the C function setColsKind which calls the GLPK function glp_set_col_kind.

Value
   NULL

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
setColsNamesGLPK

References
Based on the package glpk by Lopaka Lee.

See Also

glpkConstants

setColsNamesGLPK       Set/Change Column Names

Description
This is an advanced version of setColNameGLPK. Here, j can be an integer vector, cnames can be a character vector.

Usage

setColsNamesGLPK(lp, j, cnames = NULL)

Arguments

lp       An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
j       Vector of column numbers.
cnames       Vector of column names of the same length as j or NULL.

Details
Interface to the C function setColsNames which calls the GLPK function glp_set_col_name.
If cnames is set to NULL, all column names for column indices in j will be removed. If cname[k] is the empty string "", column name j[k] will be removed.

Value

NULL

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.
Description

Low level interface function to the GLPK function glp_set_col_stat. Consult the GLPK documentation for more detailed information.

Usage

```r
setColStatGLPK(lp, j, stat)
```

Arguments

- **lp**: An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **j**: Column number j.
- **stat**: A status parameter, see `glpkConstants`, section ‘LP/MIP problem object’ for possible values.

Details

Interface to the C function `setColStat` which calls the GLPK function `glp_set_col_stat`.

Value

`NULL`

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.

See Also

`glpkConstants`
setDefaultIptParmGLPK  Sets the Default Control Parameters for the Interior-point Method.

Description

Initializes a new structure glp_iptcp. Consult the GLPK documentation for more detailed information.

Usage

setDefaultIptParmGLPK()

Details

Interface to the C function setDefaultIptParm which calls the GLPK function glp_init_iptcp.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘Control Parameters’.

setDefaultMIPParmGLPK  Sets the Default Control Parameters for the MIP Method

Description

Initializes a new structure glp_iocp. Consult the GLPK documentation for more detailed information.

Usage

setDefaultMIPParmGLPK()
setDefaultSmpParmGLPK

Details
Interface to the C function setDefaultMIPParm which calls the GLPK function glp_init_iocp.

Value
NULL

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

See Also
glpkConstants, section ‘Control Parameters’.

setDefaultSmpParmGLPK  Sets the Default Control Parameters for the Simplex Methods.

Description
Initializes a new structure glp_smcp. Consult the GLPK documentation for more detailed information.

Usage
setDefaultSmpParmGLPK()

Details
Interface to the C function setDefaultSmpParm which calls the GLPK function glp_init_smcp.

Value
NULL

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘Control Parameters’.

Description

Sets/Changes the values of corresponding members of in the structure glp_iptcp. Consult the
GLPK documentation for more detailed information.

Usage

setInteriorParmGLPK(parm, val)

Arguments

parm        A vector containing integer values or symbolic names of the control parameters
to be changed (see glpkConstants, section ‘Control Parameters’) and ‘interior-
point solver control parameters’).
val         A vector containing the new values for the corresponding control parameters.

Details

The Arguments parm and val must have the same length. The value val[i] belongs to the param-
eter parm[i].

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
setMatColGLPK

See Also

glpkConstants

---

setMatColGLPK  Set (Replace) Column of the Constraint Matrix

Description

Low level interface function to the GLPK function glp_set_mat_col. Consult the GLPK documentation for more detailed information.

Usage

setMatColGLPK(lp, j, len, ind, val)

Arguments

- **lp**: An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
- **j**: Replace the j-th column of the constraint matrix of the specified problem object.
- **len**: Number of new column elements.
- **ind**: Row indices of the new column elements.
- **val**: Numerical values of the new column elements.

Details

Interface to the C function setMatCol which calls the GLPK function glp_set_mat_col.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
setMatRowGLPK

Set (Replace) Row of the Constraint Matrix

Description

Low level interface function to the GLPK function glp_set_mat_row. Consult the GLPK documentation for more detailed information.

Usage

setMatRowGLPK(lp, i, len, ind, val)

Arguments

- **lp**: An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
- **i**: Replace the i-th row of the constraint matrix of the specified problem object.
- **len**: Number of new row elements.
- **ind**: Column indices of the new row elements.
- **val**: Numerical values of the new row elements.

Details

Interface to the C function setMatRow which calls the GLPK function glp_set_mat_row.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
setMIPParmGLPK

Sets/Changes Control Parameters or the MIP Methods

Description

Sets/Changes the values of corresponding members of in the structure glp_iocp. Consult the GLPK documentation for more detailed information.

Usage

setMIPParmGLPK(parm, val)

Arguments

parm A vector containing integer values or symbolic names of the control parameters to be changed (see glpkConstants, section ‘Control Parameters’ and ‘integer optimizer control parameters’).

val A vector containing the new values for the corresponding control parameters.

Details

The Arguments parm and val must have the same length. The value val[i] belongs to the parameter parm[i].

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants
**setObjCoefGLPK**

*Set/Change Objective Coefficient or Constant Term*

---

**Description**

Low level interface function to the GLPK function `glp_set_obj_coef`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
setObjCoefGLPK(lp, j, obj_coef)
```

**Arguments**

- `lp`: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `j`: Column number j.
- `obj_coef`: Objective coefficient or constant term.

**Details**

Interface to the C function `setobjCoef` which calls the GLPK function `glp_set_obj_coef`.

**Value**

`NULL`

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

**setObjCoefsGLPK**

*Set/Change Objective Coefficients and/or Constant Term*

**Description**

This is an advanced version of `setColBndGLPK`. Here, `j` can be an integer vector, `obj_coef` can be a numeric vector.

**Usage**

```r
setObjCoefsGLPK(lp, j, obj_coef)
```

**Arguments**

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `j` Vector of column numbers.
- `obj_coef` Vector of objective coefficients.

**Details**

Interface to the C function `setObjCoefs` which calls the GLPK function `glp_set_obj_coef`.

**Value**

`NULL`

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

Description

Low level interface function to the GLPK function `glp_set_obj_dir`. Consult the GLPK documentation for more detailed information.

Usage

```
setObjDirGLPK(lp, lpdirt)
```

Arguments

- `lp`: An object of class "`glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `lpdir`: Optimization direction flag, which can be `GLP_MIN` (default) or `GLP_MAX`.

Details

Interface to the C function `setobjdir` which calls the GLPK function `glp_set_obj_dir`.

Value

`NULL`

Author(s)

Gabriel Gelius-Dietrich<geliedie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.

See Also

`glpkConstants`, section ‘LP/MIP problem object’.  

---
setObjNameGLPK

Set/Change Objective Function Name

Description

Low level interface function to the GLPK function \texttt{glp\_set\_obj\_name}. Consult the GLPK documentation for more detailed information.

Usage

\begin{verbatim}
setObjNameGLPK(lp, oname = NULL)
\end{verbatim}

Arguments

\begin{itemize}
  \item \texttt{lp} An object of class \texttt{glpkPtr} as returned by \texttt{initProbGLPK}. This is basically a pointer to a GLPK problem object.
  \item \texttt{oname} Objective Function name.
\end{itemize}

Details

Interface to the C function \texttt{setObjName} which calls the GLPK function \texttt{glp\_set\_obj\_name}.

Value

\texttt{NULL}

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package \texttt{glpk} by Lopaka Lee.
The GNU GLPK home page at \url{http://www.gnu.org/software/glpk/glpk.html}. 
setProbNameGLPK

Description

Low level interface function to the GLPK function glp_set_prob_name. Consult the GLPK documentation for more detailed information.

Usage

setProbNameGLPK(lp, pname = NULL)

Arguments

- lp: An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
- pname: Problem name.

Details

Interface to the C function setProbName which calls the GLPK function glp_set_prob_name.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.
**setRhsZeroGLPK**

Set/Change all Row Bounds to Zero

**Description**

This is an advanced version of `setRowsBndsGLPK`.

**Usage**

```r
setRhsZeroGLPK(lp)
```

**Arguments**

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.

**Details**

Interface to the C function `setRowsBnds` which calls the GLPK function `glp_set_col_bnds`. All row bounds are fixed at zero.

**Value**

`NULL`

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package `glpk` by Lopaka Lee.

**setRiiGLPK**

Set row scale factor

**Description**

Low level interface function to the GLPK function `glp_set_rii`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
setRiiGLPK(lp, i, rii)
```
setRowBndGLPK

Arguments
lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
i Row number i.
rii Scale factor $r_{ii}$.

Details
Interface to the C function setRii which calls the GLPK function glp_set_rii.

Value
NULL

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

setRowBndGLPK | Set/Change Row Bounds

Description
Low level interface function to the GLPK function glp_set_row_bnds. Consult the GLPK documentation for more detailed information.

Usage
setRowBndGLPK(lp, i, type, lb, ub)

Arguments
lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
i Row number i.
type Row type. For possible values, see glpkConstants, section ‘LP/MIP problem object’.
lb Lower bound.
ub Upper bound.
Details

Interface to the C function `setRowBnd` which calls the GLPK function `glp_set_row_bnds`.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.

See Also

`glpkConstants`

---

**setRowNameGLPK**  
*Set/Change Row Name*

Description

Low level interface function to the GLPK function `glp_set_row_name`. Consult the GLPK documentation for more detailed information.

Usage

`setRowNameGLPK(lp, i, rname = NULL)`

Arguments

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `i` Row number i.
- `rname` Row name.

Details

Interface to the C function `setRowName` which calls the GLPK function `glp_set_row_name`.

Value

NULL
**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package **glpk** by Lopaka Lee.

---

**setRowsBndsGLPK**

*Set/Change Row Bounds*

**Description**

This is an advanced version of **setRowBndGLPK**. Here, i can be an integer vector, lb and ub can be numeric vectors.

**Usage**

```r
setRowsBndsGLPK(lp, i, lb, ub, type = NULL)
```

**Arguments**

- `lp`: An object of class "glpkPtr" as returned by **initProbGLPK**. This is basically a pointer to a GLPK problem object.
- `i`: Vector of row numbers.
- `lb`: Vector of lower bounds.
- `ub`: Vector of upper bounds.
- `type`: Vector of variable types (default: NULL). For possible values, see **glpkConstants**, section 'LP/MIP problem object'.

**Details**

Interface to the C function setRowsBnds which calls the GLPK function glp_set_row_bnds.

If `type` is set to NULL, the type of the variables will be estimated. If `lb[j]` equals `ub[j]`, variable `i[j]` is fixed, otherwise double bounded.

**Value**

NULL

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants

---

**setRowsNamesGLPK**  
*Set/Change Row Names*

### Description

This is an advanced version of `setRowNameGLPK`. Here, i can be an integer vector, rnames can be a character vector.

### Usage

```r
setRowsNamesGLPK(lp, i, rnames = NULL)
```

### Arguments

- **lp**
  - An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **i**
  - Vector of row numbers.
- **rnames**
  - Vector of row names of the same length as i or NULL.

### Details

Interface to the C function `setRowsNames` which calls the GLPK function `glp_set_row_name`. If rnames is set to NULL, all row names for row indices in i will be removed. If `rname[k]` is the empty string "", row name i[k] will be removed.

### Value

NULL

### Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

### References

Based on the package glpk by Lopaka Lee.
Description

Low level interface function to the GLPK function glp_set_row_stat. Consult the GLPK documentation for more detailed information.

Usage

```
setRowStatGLPK(lp, i, stat)
```

Arguments

- **lp**: An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- **i**: Row number i.
- **stat**: A status parameter, see `glpkConstants` for possible values.

Details

Interface to the C function `setRowStat` which calls the GLPK function `glp_set_row_stat`, section ‘LP/MIP problem object’.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.

See Also

`glpkConstants`
setSimplexParmGLPK  

Sets/Changes Control Parameters or the Simplex Methods.

Description

Sets/Changes the values of corresponding members of in the structure glp_smcp. Consult the GLPK documentation for more detailed information.

Usage

```r
setSimplexParmGLPK(parm, val)
```

Arguments

- `parm` A vector containing integer values or symbolic names of the control parameters to be changed (see `glpkConstants`, section ‘Control Parameters’ and ‘simplex method control parameters’).
- `val` A vector containing the new values for the corresponding control parameters.

Details

The Arguments `parm` and `val` must have the same length. The value `val[i]` belongs to the parameter `parm[i]`.

Value

`NULL`

Author(s)

- Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
- Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.


See Also

- `glpkConstants`
setSjjGLPK

Retrieves column scale factor

Description

Low level interface function to the GLPK function `glp_set_sjj`. Consult the GLPK documentation for more detailed information.

Usage

```r
setSjjGLPK(lp, j, sjj)
```

Arguments

- `lp` An object of class "glpkPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `j` Column number `j`.
- `sjj` Scale factor `$s_{jj}$`.

Details

Interface to the C function `setSjj` which calls the GLPK function `glp_set_sjj`.

Value

`NULL`

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package `glpk` by Lopaka Lee.
solveInteriorGLPK  Solve LP Problem with the Interior-Point Method

Description

Low level interface function to the GLPK function glp_interior. Consult the GLPK documentation for more detailed information.

Usage

solveInteriorGLPK(lp)

Arguments

lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function solveInterior which calls the GLPK function glp_interior.

Value

A return code.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘return codes’ and return_codeGLPK.
Solve MIP Problem with the Branch-and-Cut Method

Description

Low level interface function to the GLPK function glp_intopt. Consult the GLPK documentation for more detailed information.

Usage

solveMIPGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function solveMIP which calls the GLPK function glp_intopt.

Value

A return code.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘return codes’ and return_codeGLPK.
solveSimplexExactGLPK  Solve LP Problem in Exact Arithmetic

Description

Low level interface function to the GLPK function glp_exact. Consult the GLPK documentation for more detailed information.

Usage

solveSimplexExactGLPK(lp)

Arguments

lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function solveSimplexExact which calls the GLPK function glp_exact.

Value

A return code.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘return codes’ and return_codeGLPK.
solveSimplexGLPK  Solve LP Problem with the Primal or Dual Simplex Method

Description

Low level interface function to the GLPK function glp_simplex. Consult the GLPK documentation for more detailed information.

Usage

solveSimplexGLPK(lp)

Arguments

lp  An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function solveSimplex which calls the GLPK function glp_simplex.

Value

A return code.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘return codes’ and return_codeGLPK.
sortMatrixGLPK  
*Sort Elements of the Constraint Matrix*

**Description**

Low level interface function to the GLPK function glp_sort_matrix. Consult the GLPK documentation for more detailed information.

**Usage**

sortMatrixGLPK(lp)

**Arguments**

| lp | An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object. |

**Details**

Interface to the C function sortMatrix which calls the GLPK function glp_sort_matrix.

**Value**

NULL

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>

Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package glpk by Lopaka Lee.


---

status_codeGLPK  
*Translates a GLPK Status Value into a Human Readable String*

**Description**

Translates a GLPK status code into a human readable string.

**Usage**

status_codeGLPK(code)
Arguments

code Status code from GLPK.

Value

A character string associated with the GLPK status code.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

glpkConstants, section ‘LP/MIP problem object’.

---

stdBasisGLPK Contract Standard Initial LP Basis

Description

Low level interface function to the GLPK function glp_std_basis. Consult the GLPK documentation for more detailed information.

Usage

stdBasisGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function stdBasis which calls the GLPK function glp_std_basis.

Value

NULL
Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

termOutGLPK | Enable/Disable Terminal Output

Description
Low level interface function to the GLPK function glp_term_out. Consult the GLPK documentation for more detailed information.

Usage
```
termOutGLPK(flag)
```

Arguments
flag | GLPK enable/disable flag: GLP_ON or GLP_OFF.

Details
Interface to the C function termOut which calls the GLPK function glp_term_out.

Value
Returns the previous value of the terminal output flag.

Author(s)
Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References
Based on the package glpk by Lopaka Lee.

See Also
glpkConstants, section ‘enable/disable flag’.
unscaleProbGLPK

Problem unscaling

Description

Low level interface function to the GLPK function glp_unscale_prob. Consult the GLPK documentation for more detailed information.

Usage

unscaleProbGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function unscaleProb which calls the GLPK function glp_unscale_prob.

Value

NULL

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

versionGLPK

Determine GLPK Callable Library Version

Description

Low level interface function to the GLPK function glp_version. Consult the GLPK documentation for more detailed information.

Usage

versionGLPK()
Details

Interface to the C function version which calls the GLPK function glp_version.

Value

Returns a single character value containing the GLPK version number.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

warmUpGLPK

Warm Up LP Basis

Description

Low level interface function to the GLPK function glp_warm_up. Consult the GLPK documentation for more detailed information.

Usage

warmUpGLPK(lp)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.

Details

Interface to the C function warmUp which calls the GLPK function glp_warm_up.

Value

Status of “warming up”.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>
**writeIptGLPK**  
**Write Interior-Point Solution to Text File**

**Description**

Low level interface function to the GLPK function `glp_write_ipt`. Consult the GLPK documentation for more detailed information.

**Usage**

```r
writeIptGLPK(lp, fname)
```

**Arguments**

- `lp` An object of class "glpPtr" as returned by `initProbGLPK`. This is basically a pointer to a GLPK problem object.
- `fname` The name of the text file to be written out.

**Details**

Interface to the C function `writeIpt` which calls the GLPK function `glp_write_ipt`.

**Value**

Returns zero on success, otherwise it returns non-zero.

**Author(s)**

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

**References**

Based on the package **glpk** by Lopaka Lee.

**See Also**

`printSolGLPK`, `readSolGLPK`, `writeSolGLPK`, `printIptGLPK`, `readIptGLPK`, `printMIPGLPK`, `readMIPGLPK`, and `writeMIPGLPK`. 
Write Problem Data in CPLEX LP Format

Description

Low level interface function to the GLPK function glp_write_lp. Consult the GLPK documentation for more detailed information.

Usage

writeLPGLPK(lp, fname)

Arguments

lp An object of class "glpPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fname The name of the text file to be written out.

Details

Interface to the C function writeLP which calls the GLPK function glp_write_lp.

Value

Returns zero on success, otherwise it returns non-zero and prints an error message.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

readMPSGLPK, readLPGLPK, readProbGLPK, writeMPSGLPK and writeProbGLPK.
writeMIPGLPK      Write MIP Solution to Text File

Description

Low level interface function to the GLPK function glp_write_mip. Consult the GLPK documentation for more detailed information.

Usage

 writeMIPGLPK(lp, fname)

Arguments

lp
An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fname
The name of the text file to be written out.

Details

Interface to the C function writeMIP which calls the GLPK function glp_write_mip.

Value

Returns zero on success, otherwise it returns non-zero.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

printSolGLPK, readSolGLPK, writeSolGLPK, printIptGLPK, readIptGLPK, writeIptGLPK, printMIPGLPK
and readMIPGLPK.
writeMPSGLPK  Write Problem Data in MPS Format

Description

Low level interface function to the GLPK function glp_write_mps. Consult the GLPK documentation for more detailed information.

Usage

writeMPSGLPK(lp, fmt, fname)

Arguments

lp       An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fmt      MPS format. See glpkConstants, section ‘MPS file formats’.  
fname    The name of the text file to be written out.

Details

Interface to the C function writeMPS which calls the GLPK function glp_write_mps.

Value

Returns zero on success, otherwise it returns non-zero and prints an error message.

Author(s)

Gabriel Gelius-Dietrich < Geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger < mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

readMPSGLPK, readLGPLPK, readProbGLPK, writeLGPLPK, writeProbGLPK and glpkConstants.
writeProbGLPK

Write Problem Data in GLPK Format

Description

Low level interface function to the GLPK function glp_write_prob. Consult the GLPK documentation for more detailed information.

Usage

writeProbGLPK(lp, fname)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fname The name of the text file to be written out.

Details

Interface to the C function writeProb which calls the GLPK function glp_write_prob.

Value

Returns zero on success, otherwise it returns non-zero and prints an error message.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

readMPSGLPK, readLPGLPK, readProbGLPK, writeLPGLPK, writeMPSGLPK and writeMPSGLPK.
writeSolGLPK

Write Basic Solution to Text File

Description

Low level interface function to the GLPK function glp_write_sol. Consult the GLPK documentation for more detailed information.

Usage

writeSolGLPK(lp, fname)

Arguments

lp An object of class "glpkPtr" as returned by initProbGLPK. This is basically a pointer to a GLPK problem object.
fname The name of the text file to be written out.

Details

Interface to the C function writeSol which calls the GLPK function glp_write_sol.

Value

Returns zero on success, otherwise it returns non-zero.

Author(s)

Gabriel Gelius-Dietrich <geliudie@uni-duesseldorf.de>
Maintainer: Mayo Roettger <mayo.roettger@hhu.de>

References

Based on the package glpk by Lopaka Lee.

See Also

printSolGLPK, readSolGLPK, printIptGLPK, readIptGLPK, writeIptGLPK, printMIPGLPK, readMIPGLPK and writeMIPGLPK.
Index

*Topic optimize

- addColsGLPK, 6
- addRowsGLPK, 7
- advBasisGLPK, 8
- bfExistsGLPK, 9
- bfUpdatedGLPK, 10
- checkDupGLPK, 11
- copyProbGLPK, 12
- cpxBasisGLPK, 13
- createIndexGLPK, 13
- delColsGLPK, 14
- deleteIndexGLPK, 15
- delProbGLPK, 16
- delRowsGLPK, 16
- eraseProbGLPK, 17
- factorizeGLPK, 18
- findColGLPK, 19
- findRowGLPK, 20
- getBfcpglpk, 21
- getBheadglpk, 22
- getCbindglpk, 23
- getColDualglpk, 24
- getColDualIptGLPK, 25
- getColKindGLPK, 26
- getColLowBndGLPK, 27
- getColNameGLPK, 28
- getColPrimGLPK, 29
- getColPrimIptGLPK, 30
- getColsDualGLPK, 31
- getColsDualIptGLPK, 31
- getColsKindGLPK, 32
- getColsLowBndsGLPK, 33
- getColsPrimGLPK, 34
- getColsPrimIptGLPK, 34
- getColsStatGLPK, 35
- getColStatGLPK, 36
- getColsUppBndsGLPK, 37
- getColTypeGLPK, 38
- getColUppBndGLPK, 39
- getDualStatGLPK, 40
- getInteriorParmGLPK, 41
- getMatColGLPK, 42
- getMatRowGLPK, 43
- getMIPParmGLPK, 44
- getNumBinGLPK, 45
- getNumColsGLPK, 45
- getNumIntGLPK, 46
- getNumNnzGLPK, 47
- getNumRowsGLPK, 48
- getObjCoeffGLPK, 48
- getObjCoefsGLPK, 49
- getObjDirGLPK, 50
- getObjNameGLPK, 51
- getObjValueGLPK, 52
- getObjValueIptGLPK, 52
- getPrimStatGLPK, 53
- getProbNameGLPK, 54
- getRbindGLPK, 55
- getRiiglpk, 56
- getRowDualGLPK, 57
- getRowDualIptGLPK, 58
- getRowLowBndGLPK, 59
- getRowNameGLPK, 60
- getRowPrimGLPK, 61
- getRowPrimIptGLPK, 62
- getRowsDualGLPK, 63
- getRowsDualIptGLPK, 63
- getRowsLowBndsGLPK, 64
- getRowsPrimGLPK, 65
- getRowsPrimIptGLPK, 66
- getRowsStatGLPK, 66
- getRowsTypesGLPK, 67
- getRowsUppBndsGLPK, 69
- getRowTypeGLPK, 70
- getRowUppBndGLPK, 71
- getSimplexParmGLPK, 72
- getSjjGLPK, 73
INDEX

getSolStatGLPK, 74
getSolStatIptGLPK, 75
getUnbndRayGLPK, 76
glpkAPI-package, 5
glpkConstants, 76
glpkPtr-class, 83
initProbGLPK, 84
loadMatrixGLPK, 85
mipColsValGLPK, 86
mipColValGLPK, 87
mipRowsValGLPK, 88
mipRowValGLPK, 89
mipStatusGLPK, 90
mplAllocWkspGLPK, 90
mplBuildProbGLPK, 91
mplFreeWkspGLPK, 92
mplGenerateGLPK, 93
mplPostsolveGLPK, 94
mplReadDataGLPK, 95
mplReadModelGLPK, 96
printIptGLPK, 97
printMIPGLPK, 98
printRangesGLPK, 99
printSolGLPK, 100
readIptGLPK, 101
readLPGLPK, 102
readMIPGLPK, 103
readMPSGLPK, 104
readProbGLPK, 105
readSolGLPK, 106
return_codeGLPK, 107
scaleProbGLPK, 107
setBfcpglPK, 108
setColBndGLPK, 109
setColKindGLPK, 110
setColNameGLPK, 111
setColsBndsGLPK, 112
setColsBndsObjCoefsGLPK, 113
setColsKindGLPK, 114
setColsNamesGLPK, 115
setColStatGLPK, 116
setDefaultIptParmGLPK, 117
setDefaultMIPParmGLPK, 117
setDefaultSmpParmGLPK, 118
setInteriorParmGLPK, 119
setMatColGLPK, 120
setMatRowGLPK, 121
setMIPParmGLPK, 122
setObjCoefGLPK, 123
setObjCoefsGLPK, 124
setObjDirGLPK, 125
setObjNameGLPK, 126
setProbNameGLPK, 127
setRhsZeroGLPK, 128
setRiiGLPK, 128
setRowBndsGLPK, 129
setRowNameGLPK, 130
setRowsBndsGLPK, 131
setRowsNamesGLPK, 132
setRowStatGLPK, 133
setSimplexParmGLPK, 134
setSjjGLPK, 135
solveInteriorGLPK, 136
solveMIPGLPK, 137
solveSimplexExactGLPK, 138
solveSimplexGLPK, 139
sortMatrixGLPK, 140
status_codeGLPK, 140
stdBasisGLPK, 141
termOutGLPK, 142
unscaleProbGLPK, 143
versionGLPK, 143
warmUpGLPK, 144
writeIptGLPK, 145
writeLPGLPK, 146
writeMIPGLPK, 147
writeMPSGLPK, 148
writeProbGLPK, 149
writeSolGLPK, 150

*Topic package
  glpkAPI-package, 5

addColsGLPK, 6
addRowsGLPK, 7
advBasisGLPK, 8

bfExistsGLPK, 9
bfUpdatedGLPK, 10
BINARIZE (glpkConstants), 76
BR_TECH (glpkConstants), 76
BT_TECH (glpkConstants), 76

CB_FUNC (glpkConstants), 76
CB_SIZE (glpkConstants), 76
checkDupGLPK, 11
CLQ_CUTS (glpkConstants), 76
INDEX

constantsGLPK (glpkConstants), 76

copyProbGLPK, 12

COV_CUTS (glpkConstants), 76
cpxBasisGLPK, 13

createIndexGLPK, 13, 19, 20
delColsGLPK, 14
deleteIndexGLPK, 15
delProbGLPK, 16
delRowsGLPK, 16

EPS_TOL (glpkConstants), 76
eraseProbGLPK, 17

factorizeGLPK, 18
findColGLPK, 19
findRowGLPK, 20

FP_HEUR (glpkConstants), 76

getBfcpglpk, 21
getBheadGLPK, 22
getCbindGLPK, 23
getColDualGLPK, 24, 31
getColDualIpGLPK, 25, 31
getColKindGLPK, 26, 32
getColLowBndGLPK, 27, 33
getColNameGLPK, 28
getColPrimGLPK, 29, 34
getColPrimIpGLPK, 30
getColsDualGLPK, 31
getColsDualIpGLPK, 31
getColsKindGLPK, 32
getColsLowBndGLPK, 33
getColsPrimGLPK, 34
getColsPrimIpGLPK, 34
getColsStatGLPK, 35
getColStatGLPK, 35, 36
getColsUpbndsGLPK, 37
getColTypeGLPK, 38
getColUpbndsGLPK, 37, 39
getDualStatGLPK, 40
getInteriorParmGLPK, 41
getMatColGLPK, 42
getMatRowGLPK, 43
getMIPParmGLPK, 44
getNumBinGLPK, 45
getNumColsGLPK, 45
getNumIntGLPK, 46
getNumNnzGLPK, 47

getNumRowsGLPK, 48

getObjCoefGLPK, 48, 49
getObjCoefsGLPK, 49
getObjDirGLPK, 50
getObjNameGLPK, 51
getObjValGLPK, 52
getObjValIptGLPK, 52
getPrimStatGLPK, 53

getProbNameGLPK, 54
gerbbindGLPK, 55

gerbILPK, 56

getRowDualGLPK, 57, 63

gerowDualIpGLPK, 58, 63

getRowLowBndGLPK, 59, 64

gerowNameGLPK, 60

getRowPrimGLPK, 61, 65

gerowPrimIpGLPK, 62, 66

getRowDualGLPK, 63

getRowDualIpGLPK, 63

getRowLowBndGLPK, 64

gerowPrimGLPK, 65

getRowPrimIpGLPK, 66

getRowStatGLPK, 66

getRowStatGLPK, 66, 67

gerowTypesGLPK, 68

getRowUppbndsGLPK, 69

gerowTypeGLPK, 68, 70

getRowUppbndsGLPK, 69, 71

getSimplexParmGLPK, 72

gsllGLPK, 73

getSolStatGLPK, 74

getSolStatIpGLPK, 75

getUnbndRayGLPK, 76

glp_add_cols (addColsGLPK), 6

glp_add_rows (addRowsGLPK), 7

glp_adv_basis (advBasisGLPK), 8

GLP_BF_BG (glpkConstants), 76

GLP_BF_BTF (glpkConstants), 76

glp_bf_exists (bfExistsGLPK), 9

GLP_BF_FT (glpkConstants), 76

GLP_BF_GR (glpkConstants), 76

GLP_BF_LUF (glpkConstants), 76

glp_bf_updated (bfUpdatedGLPK), 10

GLP_BR_DTH (glpkConstants), 76

GLP_BR_FFV (glpkConstants), 76

GLP_BR_LFV (glpkConstants), 76

GLP_BR_MFV (glpkConstants), 76

GLP_BR_PCH (glpkConstants), 76
INDEX

GLP_BS (glpkConstants), 76
GLP_BT_BFS (glpkConstants), 76
GLP_BT_BLB (glpkConstants), 76
GLP_BT_BPH (glpkConstants), 76
GLP_BT_DFS (glpkConstants), 76
GLP_BV (glpkConstants), 76
glp_check_dup (checkDupGLPK), 11
glp_copy_prob (copyProbGLPK), 12
glp_cpx_basis (cpxBasisGLPK), 13
glp_create_index (createIndexGLPK), 13
glp_create_prob (initProbGLPK), 84
GLP_CV (glpkConstants), 76
GLP_DB (glpkConstants), 76
glp_del_cols (delColsGLPK), 14
glp_del_rows (delRowsGLPK), 16
glp_delete_index (deleteIndexGLPK), 15
glp_delete_prob (delProbGLPK), 16
GLP_DN_BRANCH (glpkConstants), 76
GLP_DUAL (glpkConstants), 76
GLP_DUALP (glpkConstants), 76
GLP_EBADB (glpkConstants), 76
GLP_EBOUND (glpkConstants), 76
GLP_ECND (glpkConstants), 76
GLP_EDATA (glpkConstants), 76
GLP_EFAIL (glpkConstants), 76
GLP_EINSTAB (glpkConstants), 76
GLP_EITLM (glpkConstants), 76
GLP_EMIGAP (glpkConstants), 76
GLP_ENOCVG (glpkConstants), 76
GLP_ENODDFS (glpkConstants), 76
GLP_ENOFEAS (glpkConstants), 76
GLP_ENOPFS (glpkConstants), 76
GLP_EOBJLL (glpkConstants), 76
GLP_EOBJUL (glpkConstants), 76
GLP_ERANGE (glpkConstants), 76
glp_erase_prob (eraseProbGLPK), 17
GLP_EROOT (glpkConstants), 76
GLP_ESING (glpkConstants), 76
GLP_ESTOP (glpkConstants), 76
GLP_ETMLM (glpkConstants), 76
glp_exact (solveSimplexExactGLPK), 138
glp_factorize (factorizeGLPK), 18
GLP_FEAS (glpkConstants), 76
glp_find_col (findColGLPK), 19
glp_find_row (findRowGLPK), 20
GLP_FR (glpkConstants), 76
GLP_FX (glpkConstants), 76
glp_get_bfcp (getBfcpGLPK), 21
glp_get_bhead (getBheadGLPK), 22
glp_get_col_bind (getColBindGLPK), 23
glp_get_col_dual (getColDualGLPK), 24
glp_get_col_kind (getColKindGLPK), 26
glp_get_col_lb (getColLowBndGLPK), 27
glp_get_col_name (getColNameGLPK), 28
glp_get_col_prim (getColPrimGLPK), 29
glp_get_col_stat (getColStatGLPK), 36
glp_get_col_type (getColTypeGLPK), 38
glp_get_col_ub (getColUpperBndGLPK), 39
glp_get_dual_stat (getDualStatGLPK), 40
glp_get_mat_col (getMatColGLPK), 42
glp_get_mat_row (getMatRowGLPK), 43
glp_get_num_bin (getNumBinGLPK), 45
glp_get_num_cols (getNumColsGLPK), 45
glp_get_num_int (getNumIntGLPK), 46
glp_get_num_nz (getNumNzGLPK), 47
glp_get_num_rows (getNumRowsGLPK), 48
glp_get_obj_coef (getObjCoefGLPK), 48
glp_get_obj_dir (getObjDirGLPK), 50
glp_get_obj_name (getObjNameGLPK), 51
glp_get_obj_val (getObjValGLPK), 52
glp_getprim_stat (getPrimStatGLPK), 53
glp_get_prob_name (getProbNameGLPK), 54
glp_get_ril (getRilGLPK), 56
glp_get_row_bind (getRowBindGLPK), 55
glp_get_row_dual (getRowDualGLPK), 57
glp_get_row_lb (getRowLowBndGLPK), 59
glp_get_row_name (getRowNameGLPK), 60
glp_get_row_prim (getRowPrimGLPK), 61
glp_get_row_stat (getRowStatGLPK), 67
glp_get_row_type (getRowTypeGLPK), 70
glp_get_row_ub (getRowUpperBndGLPK), 71
glp_get_sjj (getSjjGLPK), 73
glp_get_status (getSolvStatusGLPK), 74
glp_get_unbnd_ray (getUnbndRayGLPK), 76
GLP_IBINGO (glpkConstants), 76
GLP_IBRANCH (glpkConstants), 76
GLP_ICUTGEN (glpkConstants), 76
GLP_IHEUR (glpkConstants), 76
GLP_INFEAS (glpkConstants), 76
glp_init_iocp (setDefaultMIPParmGLPK), 117
glp_init_iptcp (setDefaultIptParmGLPK), 117
glp_interior (solveInteriorGLPK), 136
glp_intopt (solveMIPGLPK), 137
GLP_IPREPRO (glpkConstants), 76
INDEX

GLP_IPT (glpkConstants), 76
glp_ipt_col_dual (getColDualIptGLPK), 25
glp_ipt_col_prim (getColPrimIptGLPK), 30
glp_ipt_obj_val (getObjValIptGLPK), 52
glp_ipt_row_dual (getRowDualIptGLPK), 58
glp_ipt_row_prim (getRowPrimIptGLPK), 62
glp_ipt_status (getSolvStatIptGLPK), 75
GLP_IROWGEN (glpkConstants), 76
GLP_ISELECT (glpkConstants), 76
GLP_IV (glpkConstants), 76
GLP_KKT_CS (glpkConstants), 76
GLP_KKT_DB (glpkConstants), 76
GLP_KKT_DE (glpkConstants), 76
GLP_KKT_PB (glpkConstants), 76
GLP_KKT_PE (glpkConstants), 76
GLP_LO (glpkConstants), 76
glp_load_matrix (loadMatrixGLPK), 85
GLP_MAX (glpkConstants), 76
GLP_MIN (glpkConstants), 76
GLP_MIP (glpkConstants), 76
glp_mip_col_val (mipColValGLPK), 86
glp_mip_obj_val (mipObjValGLPK), 87
glp_mip_row_val (mipRowValGLPK), 89
glp_mip_status (mipStatusGLPK), 90
glp_mpl_alloc_wksp (mplAllocWkspGLPK), 90
glp_mpl_build_prob (mplBuildProbGLPK), 91
glp_mpl_free_wksp (mplFreeWkspGLPK), 92
glp_mpl_generate (mplGenerateGLPK), 93
glp_mpl_postsolve (mplPostsolveGLPK), 94
glp_mpl_read_data (mplReadDataGLPK), 95
glp_mpl_read_model (mplReadModelGLPK), 96
GLP_MPS_DECK (glpkConstants), 76
GLP_MPS_FILE (glpkConstants), 76
GLP_MSG_ALL (glpkConstants), 76
GLP_MSG_DBG (glpkConstants), 76
GLP_MSG_ERR (glpkConstants), 76
GLP_MSG_OFF (glpkConstants), 76
GLP_MSG_ON (glpkConstants), 76
GLP_NO_BRNCH (glpkConstants), 76
GLP_NOFEAS (glpkConstants), 76
GLP_NS (glpkConstants), 76
GLP_NU (glpkConstants), 76
GLP_OFF (glpkConstants), 76
GLP_ON (glpkConstants), 76
GLP_OPT (glpkConstants), 76
GLP_ORD_AMD (glpkConstants), 76
GLP_ORD_NONE (glpkConstants), 76
GLP_ORD_QMD (glpkConstants), 76
GLP_ORD_SYMAMD (glpkConstants), 76
GLP_PP_ALL (glpkConstants), 76
GLP_PP_NONE (glpkConstants), 76
GLP_PP_ROOT (glpkConstants), 76
GLP_PRIMAL (glpkConstants), 76
glp_print_ipt (printIptGLPK), 97
glp_print_mip (printMipGLPK), 98
glp_print_ranges (printRangesGLPK), 99
glp_print_sol (printSolGLPK), 100
GLP_PT_PSE (glpkConstants), 76
GLP_PT_STD (glpkConstants), 76
glp_read_ipt (readIptGLPK), 101
glp_read_lp (readLPGLPK), 102
glp_read_mip (readMIPGLPK), 103
glp_read_mps (readMPSGLPK), 104
glp_read_prob (readProbGLPK), 105
glp_read_sol (readSolGLPK), 106
GLP_RF_CLQ (glpkConstants), 76
GLP_RF_COV (glpkConstants), 76
GLP_RF_CUT (glpkConstants), 76
GLP_RF_QMI (glpkConstants), 76
GLP_RF_LAZY (glpkConstants), 76
GLP_RF_MIR (glpkConstants), 76
GLP_RF_REG (glpkConstants), 76
GLP_RT_HAR (glpkConstants), 76
GLP_RT_STD (glpkConstants), 76
glp_scale_prob (scaleProbGLPK), 107
glp_set_bfcp (setBfcpGLPK), 108
glp_set_col_bnds (setColBndGLPK), 109
glp_set_col_kind (setColKindGLPK), 110
glp_set_col_name (setColNameGLPK), 111
glp_set_col_stat (setColStatGLPK), 116
glp_set_mat_col (setMatColGLPK), 120
glp_set_mat_row (setMatRowGLPK), 121
glp_set_obj_coef (setObjCoefGLPK), 123
glp_set_obj_dir (setObjDirGLPK), 125
glp_set_obj_name (setObjNameGLPK), 126
glp_set_prob_name (setProbNameGLPK), 127
glp_set_rri (setRriGLPK), 128
glp_set_row_bnds (setRowBndGLPK), 129
glp_set_row_name (setRowNameGLPK), 130
glp_set_row_stat (setRowStatGLPK), 133
glp_set_sjj (setSjjGLPK), 135
INDEX

printIptGLPK, 97, 98, 100, 101, 103, 106, 145, 147, 150
printMIPGLPK, 97, 98, 100, 101, 103, 106, 145, 147, 150
printRangesGLPK, 99
printSolGLPK, 97, 98, 100, 101, 103, 106, 145, 147, 150
R_TEST (glpkConstants), 76
readIptGLPK, 97, 98, 100, 101, 103, 106, 145, 147, 150
readLPGLPK, 102, 104, 105, 146, 148, 149
readMIPGLPK, 97, 98, 100, 101, 103, 106, 145, 147, 150
readMPSGLPK, 102, 104, 105, 146, 148, 149
readProbGLPK, 102, 104, 105, 146, 148, 149
readSolGLPK, 97, 98, 100, 101, 103, 106, 145, 147, 150
return_codeGLPK, 83, 107, 136–139
RS_SIZE (glpkConstants), 76
scaleProbGLPK, 107
setBfcpglpk, 108
setColBndGLPK, 109, 112, 124
setColKindGLPK, 110, 114
setColNameGLPK, 111, 115
setColsBndsGLPK, 112, 113
setColsBndsObjCoefsGLPK, 113
setColsKindGLPK, 114
setColsNamesGLPK, 115
setColStatGLPK, 116
setDefaultIptParmGLPK, 117
setDefaultMIPParmGLPK, 117
setDefaultSmpParmGLPK, 118
setInteriorParmGLPK, 119
setMatColGLPK, 120
setMatRowGLPK, 121
setMIPParmGLPK, 122
setObjCoefGLPK, 123
setObjCoefsGLPK, 113, 124
setObjDirGLPK, 125
setObjNameGLPK, 126
setProbNameGLPK, 127
setRhsZeroGLPK, 128
setRiiiGLPK, 128
setRowBndGLPK, 129, 131
setRowNameGLPK, 130, 132
setRowsBndsGLPK, 128, 131
setRowsNamesGLPK, 132
setRowStatGLPK, 133
setSimplexParmGLPK, 134
setSjjGLPK, 135
solveInteriorGLPK, 136
solveMIPGLPK, 137
solveSimplexExactGLPK, 138
solveSimplexGLPK, 139
sortMatrixGLPK, 140
status_codeGLPK, 83, 140
stdBasisGLPK, 141
SUHL (glpkConstants), 76
termOutGLPK, 142
TM_LIM (glpkConstants), 76
TOL_BND (glpkConstants), 76
TOL_DJ (glpkConstants), 76
TOL_INT (glpkConstants), 76
TOL_OBJ (glpkConstants), 76
TOL_PIV (glpkConstants), 76
TYPE (glpkConstants), 76
unscaleProbGLPK, 143
UPD_TOL (glpkConstants), 76
versionGLPK, 143
warmUpGLPK, 144
writeIptGLPK, 97, 98, 100, 101, 103, 106, 145, 147, 150
writeLPGLPK, 102, 104, 105, 146, 148, 149
writeMIPGLPK, 97, 98, 100, 101, 103, 106, 145, 147, 150
writeMPSGLPK, 102, 104, 105, 146, 148, 149
writeProbGLPK, 102, 104, 105, 146, 148, 149
writeSolGLPK, 97, 98, 100, 101, 103, 106, 145, 147, 150