

NAME

CUTEST_csgreh_threaded – CUTEst tool to evaluate the constraint gradients, the Lagrangian Hessian in finite element format and the gradient of either the objective/Lagrangian in sparse format.

SYNOPSIS

CALL CUTEST_csgreh_threaded(status, n, m, X, Y, grlagf, nnzj, lj, J_val, J_var, J_fun, ne, lhe_ptr, HE_row_ptr, HE_val_ptr, lhe_row, HE_row, lhe_val, HE_val, byrows, thread)

DESCRIPTION

The CUTEST_csgreh_threaded subroutine evaluates both the gradients of the general constraint functions and the Hessian matrix of the Lagrangian function $l(x, y) = f(x) + y^T c(x)$ for the problem decoded into OUTSDIF.d at the point $(x, y) = (X, Y)$. This Hessian matrix is stored as a sparse matrix in finite element format

$$H = \sum_{e=1}^{ne} H_e,$$

where each square symmetric element H_e involves a small subset of the rows of the Hessian matrix. The subroutine also obtains the gradient of either the objective function or the Lagrangian function, stored in a sparse format.

The problem under consideration consists in minimizing (or maximizing) an objective function $f(x)$ over all $x \in R^n$ subject to general equations $c_i(x) = 0$, ($i \in 1, \dots, m_E$), general inequalities $c_i^l(x) \leq c_i(x) \leq c_i^u(x)$, ($i \in m_E + 1, \dots, m$), and simple bounds $x^l \leq x \leq x^u$. The objective function is group-partially separable and all constraint functions are partially separable.

ARGUMENTS

The arguments of CUTEST_csgreh_threaded are as follows

status [out] - integer

the output status: 0 for a successful call, 1 for an array allocation/deallocation error, 2 for an array bound error, 3 for an evaluation error, 4 for an out-of-range thread,

n [in] - integer

the number of variables for the problem,

m [in] - integer

the total number of general constraints,

X [in] - real/double precision

an array which gives the current estimate of the solution of the problem,

Y [in] - real/double precision

an array which gives the Lagrange multipliers,

grlagf [in] - logical

a logical variable which should be set .TRUE. if the gradient of the Lagrangian function is required and .FALSE. if the gradient of the objective function is sought,

nnzj [out] - integer

the number of nonzeros in J_val,

HE_row [out] - integer

an array which holds a list of the row indices involved with each element. Those for element i directly precede those for element $i+1$, $i = 1, \dots, ne-1$. Since the elements are symmetric, HE_row is also the list of column indices involved with each element.

lj [in] - integer

the actual declared dimensions of J_val, J_var and J_fun,

J_val [out] - real/double precision

an array which gives the values of the nonzeros of the gradients of the objective, or Lagrangian, and general constraint functions evaluated at X and Y. The i -th entry of J_val gives the value of the

derivative with respect to variable $J_var(i)$ of function $J_fun(i)$,

J_var [out] - integer

an array whose i -th component is the index of the variable with respect to which $J_val(i)$ is the derivative,

J_fun [out] - integer

an array whose i -th component is the index of the problem function whose value $J_val(i)$ is the derivative. $J_fun(i) = 0$ indicates the objective function whenever `grlagf` is `.FALSE.` or the Lagrangian function when `grlagf` is `.TRUE.`, while $J_fun(i) = j > 0$ indicates the j -th general constraint function.

ne [out] - integer

the number, `ne`, of "finite-elements" used,

lhe_ptr [in] - integer

the actual declared dimensions of `HE_row_ptr` and `HE_val_ptr`,

HE_row_ptr [out] - integer

`HE_row_ptr(i)` points to the position in `HE_row` of the first row index involved with element number e : the row indices of element number e are stored in `HE_row` between the indices `HE_row_ptr(e)` and `HE_row_ptr(e+1)-1`. `HE_row_ptr(ne+1)` points to the first empty location in `HE_row`,

HE_val_ptr [out] - integer

`HE_val_ptr(i)` points to the position in `HE_val` of the first nonzero involved with element number i : the values involved in element number e are stored in `HE_val` between the indices `HE_val_ptr(e)` and `HE_val_ptr(e+1)-1`. `HE_val_ptr(ne+1)` points to the first empty location in `HE_val`,

lhe_row [in] - integer

the actual declared dimension of `HE_row`,

HE_row [out] - integer

an array which holds a list of the row indices involved with each element. Those for element e directly precede those for element $e+1$, $e = 1, \dots, ne-1$. Since the elements are symmetric, `HE_row` is also the list of column indices involved with each element.

lhe_val [in] - integer

the actual declared dimension of `HE_val`,

HE_val [out] - real/double precision

an array of the nonzeros in the upper triangle of H_e , evaluated at X and stored by rows, or by columns. Those for element e directly precede those for element $e+1$, $e = 1, \dots, ne-1$. Element number e contains the values stored between

`HE_val(HE_val_ptr(e))` and `HE_val(HE_val_ptr(e+1)-1)`

and involves the rows/columns stored between

`HE_row(HE_row_ptr(e))` and `HE_row(HE_row_ptr(e+1)-1)`.

byrows [in] - logical

must be set to `.TRUE.` if the upper triangle of each H_i is to be stored by rows, and to `.FALSE.` if it is to be stored by columns,

thread [in] - integer

thread chosen for the evaluation; threads are numbered from 1 to the value `threads` set when calling `CUTEST_csetup_threaded`.

AUTHORS

I. Bongartz, A.R. Conn, N.I.M. Gould, D. Orban and Ph.L. Toint

SEE ALSO

CUTEst: a Constrained and Unconstrained Testing Environment with safe threads,
N.I.M. Gould, D. Orban and Ph.L. Toint,

Technical Report, Rutherford Appleton Laboratory, 2013.

CUTEr (and SifDec): A Constrained and Unconstrained Testing Environment, revisited,
N.I.M. Gould, D. Orban and Ph.L. Toint,
ACM TOMS, **29**:4, pp.373-394, 2003.

CUTE: Constrained and Unconstrained Testing Environment, I. Bongartz, A.R. Conn, N.I.M. Gould and
Ph.L. Toint, ACM TOMS, **21**:1, pp.123-160, 1995.

cutest_ugreh_threaded(3M), sifdecoder(1).