DuQuad

v1.0

Generated by Doxygen 1.8.6

Sat Dec 13 2014 18:15:38

Contents

1 Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

2 File Index

2.1 File List

Here is a list of all files with brief descriptions:

3 Data Structure Documentation

3.1 Array Struct Reference

`#include <qp_structs.h>`

Data Fields

- real_t * array
- uint32_t used
- uint32_t size

3.1.1 Detailed Description

for internal use

Definition at line 73 of file qp_structs.h.

3.1.2 Field Documentation

3.1.2.1 real_t* Array::array

Definition at line 74 of file qp_structs.h.

3.1.2.2 uint32_t Array::size

Definition at line 76 of file qp_structs.h.

3.1.2.3 uint32_t Array::used

Definition at line 75 of file qp_structs.h.

The documentation for this struct was generated from the following file:

- include/qp_structs.h

3.2 Info Struct Reference

`#include <qp_structs.h>`

Data Fields

- boolean lb_is_inf
- boolean ub_is_inf
- boolean lb_hat_is_inf
- boolean ub_hat_is_inf
- real_t eigH_max
- real_t eigH_min
- real_t Ld
- uint32_t problem_case
- uint32_t pf_vec_length

3.2.1 Detailed Description

parameters that are calculated automatically off-line in duquad.m

Definition at line 38 of file qp_structs.h.

3.2.2 Field Documentation

3.2.2.1 real_t Info::eigH_max

Largest eigenvalue of the Hessian H

Definition at line 43 of file qp_structs.h.

3.2.2.2 real_t Info::eigH_min

Smallest eigenvalue of the Hessian H

Definition at line 44 of file qp_structs.h.

3.2.2.3 boolean Info::lb_hat_is_inf

true: no lower bound on linear constraints

Definition at line 41 of file qp_structs.h.

3.2.2.4 boolean Info::lb_is_inf

true: no lower bound on optimization variable

Definition at line 39 of file qp_structs.h.

3.2.2.5 real_t Info::Ld

Lipschitz constant

Definition at line 45 of file qp_structs.h.

3.2.2.6 uint32_t Info::pf_vec_length

Definition at line 47 of file qp_structs.h.

3.2.2.7 uint32_t Info::problem_case

case 1: lb_hat != ub_hat, case 2: lb_hat == ub_hat, case 3: lb_hat = -inf, case 4: ub_hat = inf

Definition at line 46 of file qp_structs.h.

3.2.2.8 boolean Info::ub_hat_is_inf

true: no upper bound on linear constraint

Definition at line 42 of file qp_structs.h.

3.2.2.9 boolean Info::ub_is_inf

true: no upper bound on optimization variable

Definition at line 40 of file qp_structs.h.

The documentation for this struct was generated from the following file:

- include/qp_structs.h

3.3 Options Struct Reference

`#include <qp_structs.h>`

Data Fields

- uint32_t maxiter_outer
- uint32_t maxiter_inner
- real_t eps_ds
- real_t eps_pf
- real_t eps_inner
- uint32_t algorithm
- real_t rho

3.3.1 Detailed Description

Option specified by the user. Default values can be found in the user manual

Definition at line 28 of file qp_structs.h.

3.3.2 Field Documentation

3.3.2.1 uint32_t Options::algorithm

Specifies the algorithm used to solve the problem. Values: 1: DGM last, 2: DGM avg, 3: DFGM last, 4: DFGM avg, 5: ALM last, 6: ALM avg, 7: FALM last, 8: FALM avg

Definition at line 34 of file qp_structs.h.

3.3.2.2 real_t Options::eps_ds

Tolerance for dual suboptimality

Definition at line 31 of file qp_structs.h.

3.3.2.3 real_t Options::eps_inner

Tolerance for primal feasibility in the inner problem

Definition at line 33 of file qp_structs.h.

3.3.2.4 real_t Options::eps_pf

Tolerance for primal feasibility

Definition at line 32 of file qp_structs.h.

3.3.2.5 uint32_t Options::maxiter_inner

Maximum number of iterations in the inner loop

Definition at line 30 of file qp_structs.h.

3.3.2.6 uint32_t Options::maxiter_outer

Maximum number of iterations in the outer loop

Definition at line 29 of file qp_structs.h.

3.3.2.7 real_t Options::rho

Penalty parameter used in ALM and FALM

Definition at line 35 of file qp_structs.h.

The documentation for this struct was generated from the following file:

- include/qp_structs.h

3.4 Output Struct Reference

`#include <qp_structs.h>`

Data Fields

- uint32_t iterations
- uint32_t iterations_inner_tot
- real_t time
- real_t time_tot_inner
- uint32_t flag_last_satisfied
- uint32_t niter_feasible_ds
- uint32_t niter_feasible_pf
- uint32_t exitflag_inner
- uint32_t num_exceeded_max_niter_inner
- real_t * ds_vector
- real_t * pf_vector

3.4.1 Detailed Description

Important results are collected in the Output struct

Definition at line 50 of file qp_structs.h.

3.4.2 Field Documentation

3.4.2.1 real_t* Output::ds_vector

Vector storing all the value of the dual suboptimality every iteration

Definition at line 60 of file qp_structs.h.

3.4.2.2 uint32_t Output::exitflag_inner

Exitflag for the inner problem. Values: 1 = feasible point found, 2 = Maximum number of iterations exceeded

Definition at line 58 of file qp_structs.h.

3.4.2.3 uint32_t Output::flag_last_satisfied

Flag spesifies which stopping criteria was resolved last. Value: 0 = dual suboptimality, 1 = primal feasibility

Definition at line 55 of file qp_structs.h.

3.4.2.4 uint32_t Output::iterations

Number of outer iterations

Definition at line 51 of file qp_structs.h.

3.4.2.5 uint32_t Output::iterations_inner_tot

Total number of iterations for the inner problem

Definition at line 52 of file qp_structs.h.

3.4.2.6 uint32_t Output::niter_feasible_ds

Number of iterations the criterion for dual suboptimality was satisfied

Definition at line 56 of file qp_structs.h.

3.4.2.7 uint32_t Output::niter_feasible_pf

Number of iterations the criterion for primal feasibility was satisfied

Definition at line 57 of file qp_structs.h.

3.4.2.8 uint32_t Output::num_exceeded_max_niter_inner

Total number of times the inner problem exceeded the number of iterations

Definition at line 59 of file qp_structs.h.

3.4.2.9 real_t* Output::pf_vector

Vector storing all the value of the primal feasibility every iteration

Definition at line 61 of file qp_structs.h.

3.4.2.10 real_t Output::time

Runtime of the algorithm after all initialization is done

Definition at line 53 of file qp_structs.h.

3.4.2.11 real_t Output::time_tot_inner

Total time spent on solving the inner problem

Definition at line 54 of file qp_structs.h.

The documentation for this struct was generated from the following file:

- include/qp_structs.h

3.5 Problem Struct Reference

`#include <qp_structs.h>`

Data Fields

- real_t * H
- real_t * c
- real_t * A
- real_t * A_t
- real_t * b
- real_t * lb_hat
- real_t * ub_hat
- real_t * lb
- real_t * ub
- real_t * z0

3.5.1 Detailed Description

Contains all the matrices and vectors used to describe the general QP

Definition at line 15 of file qp_structs.h.

3.5.2 Field Documentation

3.5.2.1 real_t* Problem::A

Linear constraints matrix Dimensions m x n

Definition at line 18 of file qp_structs.h.

3.5.2.2 real_t* Problem::A_t

A transposed

Definition at line 19 of file qp_structs.h.

3.5.2.3 real_t* Problem::b

Linear constraints vector

Definition at line 20 of file qp_structs.h.

3.5.2.4 real_t* Problem::c

The gradient vector

Definition at line 17 of file qp_structs.h.

3.5.2.5 real_t* Problem::H

The Hessian matrix. Dimensions n x n, and has to be positive definite

Definition at line 16 of file qp_structs.h.

3.5.2.6 real_t* Problem::lb

The lower bound for optimization variable z

Definition at line 23 of file qp_structs.h.

3.5.2.7 real_t* Problem::lb_hat

The lower bound for the linear constraints

Definition at line 21 of file qp_structs.h.

3.5.2.8 real_t* Problem::ub

The upper bound for optimization variable z

Definition at line 24 of file qp_structs.h.

3.5.2.9 real_t* Problem::ub_hat

The upper bound for the linear constraints

Definition at line 22 of file qp_structs.h.

3.5.2.10 real_t* Problem::z0

The initial point

Definition at line 25 of file qp_structs.h.

The documentation for this struct was generated from the following file:

- include/qp_structs.h

3.6 Result Struct Reference

`#include <qp_structs.h>`

Data Fields

- real_t * zopt
- real_t fopt
- uint32_t exitflag
- real_t * lambda1
- real_t * lambda2
- struct Output * out

3.6.1 Detailed Description

Outputs of the algorithms

Definition at line 64 of file qp_structs.h.

3.6.2 Field Documentation

3.6.2.1 uint32_t Result::exitflag

Values: 1 = optimal point found, 2 = maximum number of iterations exceeded, -1 = error

Definition at line 67 of file qp_structs.h.

3.6.2.2 real_t Result::fopt

Optimal value

Definition at line 66 of file qp_structs.h.

3.6.2.3 real_t* Result::lambda1

Set of lagrangian multipliers

Definition at line 68 of file qp_structs.h.

3.6.2.4 real_t* Result::lambda2

Set of lagrangian multipliers

Definition at line 69 of file qp_structs.h.

3.6.2.5 struct Output* Result::out

Sruct containing other results

Definition at line 70 of file qp_structs.h.

3.6.2.6 real_t* Result::zopt

Optimal point

Definition at line 65 of file qp_structs.h.

The documentation for this struct was generated from the following file:

- include/qp_structs.h

3.7 Struct_ALM Struct Reference

`#include <alm.h>`

Data Fields

- struct Problem * prob
- struct Options * opt
- struct Info * info
- struct Result * res
- real_t * z
- real_t * lambda
- real_t * temp1_dim_N
- real_t * temp2_dim_M
- real_t * temp3_dim_M
- real_t * z_avg
- real_t * summ
- real_t * pf_vec
- real_t * A_z
- real_t * H_hat
- real_t * A2
- real_t * rho_At_b

3.7.1 Detailed Description

Struct containing all necessary vectors and parameters for running ALM

Definition at line 21 of file alm.h.

3.7.2 Field Documentation

3.7.2.1 real_t* Struct_ALM::A2

Definition at line 41 of file alm.h.

3.7.2.2 real_t* Struct_ALM::A_z

Definition at line 37 of file alm.h.

3.7.2.3 real_t* Struct_ALM::H_hat

Definition at line 40 of file alm.h.

3.7.2.4 struct Info* Struct_ALM::info

Definition at line 25 of file alm.h.

3.7.2.5 real_t* Struct_ALM::lambda

Definition at line 30 of file alm.h.

3.7.2.6 struct Options* Struct_ALM::opt

Definition at line 24 of file alm.h.

3.7.2.7 real_t* Struct_ALM::pf_vec

Definition at line 36 of file alm.h.

3.7.2.8 struct Problem* Struct_ALM::prob

Definition at line 23 of file alm.h.

3.7.2.9 struct Result* Struct_ALM::res

Definition at line 26 of file alm.h.

3.7.2.10 real_t* Struct_ALM::rho_At_b

Definition at line 42 of file alm.h.

3.7.2.11 real_t* Struct_ALM::summ

Definition at line 35 of file alm.h.

3.7.2.12 real_t* Struct_ALM::temp1_dim_N

Definition at line 31 of file alm.h.

3.7.2.13 real_t* Struct_ALM::temp2_dim_M

Definition at line 32 of file alm.h.

3.7.2.14 real_t* Struct_ALM::temp3_dim_M

Definition at line 33 of file alm.h.

3.7.2.15 real_t* Struct_ALM::z

Definition at line 29 of file alm.h.

3.7.2.16 real_t* Struct_ALM::z_avg

Definition at line 34 of file alm.h.

The documentation for this struct was generated from the following file:

- include/alm.h

3.8 Struct_DFGM Struct Reference

`#include <dfgm.h>`

Data Fields

- struct Problem * prob
- struct Options * opt
- struct Info * info
- struct Result * res
- real_t * z
- real_t * lambda1
- real_t * lambda2
- real_t * temp1_dim_N
- real_t * temp2_dim_M
- real_t * temp3_dim_M
- real_t * b_ub_hat
- real_t * b_lb_hat
- real_t * z_avg

- real_t * summ
- real_t * pf_vec
- real_t * A_z
- real_t * lambda1_old
- real_t * lambda2_old
- real_t * y1
- real_t * z_ds
- real_t * y2
- real_t * A_z_ds
- real_t time_inner_y
- uint32_t iterations_inner_y

3.8.1 Detailed Description

Struct containing all necessary vectors and parameters for running DFGM

Definition at line 20 of file dfgm.h.

3.8.2 Field Documentation

3.8.2.1 real_t* Struct_DFGM::A_z

Definition at line 39 of file dfgm.h.

3.8.2.2 real_t* Struct_DFGM::A_z_ds

Definition at line 47 of file dfgm.h.

3.8.2.3 real_t* Struct_DFGM::b_lb_hat

Definition at line 35 of file dfgm.h.

3.8.2.4 real_t* Struct_DFGM::b_ub_hat

Definition at line 34 of file dfgm.h.

3.8.2.5 struct Info* Struct_DFGM::info

Definition at line 24 of file dfgm.h.

3.8.2.6 uint32_t Struct_DFGM::iterations_inner_y

Definition at line 50 of file dfgm.h.

3.8.2.7 real_t* Struct_DFGM::lambda1

Definition at line 29 of file dfgm.h.

3.8.2.8 real_t* Struct_DFGM::lambda1_old

Definition at line 42 of file dfgm.h.

3.8.2.9 real_t* Struct_DFGM::lambda2

Definition at line 30 of file dfgm.h.

3.8.2.10 real_t* Struct_DFGM::lambda2_old

Definition at line 43 of file dfgm.h.

3.8.2.11 struct Options* Struct_DFGM::opt

Definition at line 23 of file dfgm.h.

3.8.2.12 real_t* Struct_DFGM::pf_vec

Definition at line 38 of file dfgm.h.

3.8.2.13 struct Problem* Struct_DFGM::prob

Definition at line 22 of file dfgm.h.

3.8.2.14 struct Result* Struct_DFGM::res

Definition at line 25 of file dfgm.h.

3.8.2.15 real_t* Struct_DFGM::summ

Definition at line 37 of file dfgm.h.

3.8.2.16 real_t* Struct_DFGM::temp1_dim_N

Definition at line 31 of file dfgm.h.

3.8.2.17 real_t* Struct_DFGM::temp2_dim_M

Definition at line 32 of file dfgm.h.

3.8.2.18 real_t* Struct_DFGM::temp3_dim_M

Definition at line 33 of file dfgm.h.

3.8.2.19 real_t Struct_DFGM::time_inner_y

Definition at line 49 of file dfgm.h.

3.8.2.20 real_t* Struct_DFGM::y1

Definition at line 44 of file dfgm.h.

3.8.2.21 real_t* Struct_DFGM::y2

Definition at line 46 of file dfgm.h.

3.8.2.22 real_t* Struct_DFGM::z

Definition at line 28 of file dfgm.h.

3.8.2.23 real_t* Struct_DFGM::z_avg

Definition at line 36 of file dfgm.h.

3.8.2.24 real_t* Struct_DFGM::z_ds

Definition at line 45 of file dfgm.h.

The documentation for this struct was generated from the following file:

- include/dfgm.h

3.9 Struct_DGM Struct Reference

`#include <dgm.h>`

Data Fields

- struct Problem * prob
- struct Options * opt
- struct Info * info
- struct Result * res
- real_t * z
- real_t * lambda1
- real_t * lambda2
- real_t * temp1_dim_N
- real_t * temp2_dim_M
- real_t * temp3_dim_M
- real_t * b_ub_hat
- real_t * b_lb_hat
- real_t * z_avg
- real_t * summ
- real_t * pf_vec
- real_t * A_z

3.9.1 Detailed Description

Struct containing all necessary vectors and parameters for running DGM

Definition at line 20 of file dgm.h.

3.9.2 Field Documentation

3.9.2.1 real_t* Struct_DGM::A_z

Definition at line 39 of file dgm.h.

3.9.2.2 real_t* Struct_DGM::b_lb_hat

Definition at line 35 of file dgm.h.

3.9.2.3 real_t* Struct_DGM::b_ub_hat

Definition at line 34 of file dgm.h.

3.9.2.4 struct Info* Struct_DGM::info

Definition at line 24 of file dgm.h.

3.9.2.5 real_t* Struct_DGM::lambda1

Definition at line 29 of file dgm.h.

3.9.2.6 real_t* Struct_DGM::lambda2

Definition at line 30 of file dgm.h.

3.9.2.7 struct Options* Struct_DGM::opt

Definition at line 23 of file dgm.h.

3.9.2.8 real_t* Struct_DGM::pf_vec

Definition at line 38 of file dgm.h.

3.9.2.9 struct Problem* Struct_DGM::prob

Definition at line 22 of file dgm.h.

3.9.2.10 struct Result* Struct_DGM::res

Definition at line 25 of file dgm.h.

3.9.2.11 real_t* Struct_DGM::summ

Definition at line 37 of file dgm.h.

3.9.2.12 real_t* Struct_DGM::temp1_dim_N

Definition at line 31 of file dgm.h.

3.9.2.13 real_t* Struct_DGM::temp2_dim_M

Definition at line 32 of file dgm.h.

3.9.2.14 real_t* Struct_DGM::temp3_dim_M

Definition at line 33 of file dgm.h.

3.9.2.15 real_t* Struct_DGM::z

Definition at line 28 of file dgm.h.

3.9.2.16 real_t* Struct_DGM::z_avg

Definition at line 36 of file dgm.h.

The documentation for this struct was generated from the following file:

- include/dgm.h

3.10 Struct_FALM Struct Reference

`#include <falm.h>`

Data Fields

- struct Problem * prob
- struct Options * opt
- struct Info * info
- struct Result * res
- real_t * z
- real_t * lambda
- real_t * temp1_dim_N
- real_t * temp2_dim_M
- real_t * temp3_dim_M
- real_t * z_avg
- real_t * summ
- real_t * pf_vec
- real_t * A_z

- real_t * lambda_old
- real_t * y1
- real_t * z_ds
- real_t * A_z_ds
- real_t time_inner_y
- uint32_t iterations_inner_y
- real_t * H_hat
- real_t * A2
- real_t * rho_At_b

3.10.1 Detailed Description

Struct containing all necessary vectors and parameters for running FALM

Definition at line 20 of file falm.h.

3.10.2 Field Documentation

3.10.2.1 real_t* Struct_FALM::A2

Definition at line 50 of file falm.h.

3.10.2.2 real_t* Struct_FALM::A_z

Definition at line 37 of file falm.h.

3.10.2.3 real_t* Struct_FALM::A_z_ds

Definition at line 43 of file falm.h.

3.10.2.4 real_t* Struct_FALM::H_hat

Definition at line 49 of file falm.h.

3.10.2.5 struct Info* Struct_FALM::info

Definition at line 24 of file falm.h.

3.10.2.6 uint32_t Struct_FALM::iterations_inner_y

Definition at line 46 of file falm.h.

3.10.2.7 real_t* Struct_FALM::lambda

Definition at line 29 of file falm.h.

3.10.2.8 real_t* Struct_FALM::lambda_old

Definition at line 40 of file falm.h.

3.10.2.9 struct Options* Struct_FALM::opt

Definition at line 23 of file falm.h.

3.10.2.10 real_t* Struct_FALM::pf_vec

Definition at line 36 of file falm.h.

3.10.2.11 struct Problem* Struct_FALM::prob

Definition at line 22 of file falm.h.

3.10.2.12 struct Result* Struct_FALM::res

Definition at line 25 of file falm.h.

3.10.2.13 real_t* Struct_FALM::rho_At_b

Definition at line 51 of file falm.h.

3.10.2.14 real_t* Struct_FALM::summ

Definition at line 35 of file falm.h.

3.10.2.15 real_t* Struct_FALM::temp1_dim_N

Definition at line 31 of file falm.h.

3.10.2.16 real_t* Struct_FALM::temp2_dim_M

Definition at line 32 of file falm.h.

3.10.2.17 real_t* Struct_FALM::temp3_dim_M

Definition at line 33 of file falm.h.

3.10.2.18 real_t Struct_FALM::time_inner_y

Definition at line 45 of file falm.h.

3.10.2.19 real_t* Struct_FALM::y1

Definition at line 41 of file falm.h.

3.10.2.20 real_t* Struct_FALM::z

Definition at line 28 of file falm.h.

3.10.2.21 real_t* Struct_FALM::z_avg

Definition at line 34 of file falm.h.

3.10.2.22 real_t* Struct_FALM::z_ds

Definition at line 42 of file falm.h.

The documentation for this struct was generated from the following file:

- include/falm.h

3.11 Struct_FGM Struct Reference

`#include <fgm.h>`

Data Fields

- real_t * H
- real_t * c

- real_t * lb
- real_t * ub
- real_t * z0
- real_t * zopt
- real_t fopt
- uint32_t exitflag
- boolean lb_is_inf
- boolean ub_is_inf
- real_t eigH_max
- real_t eigH_min
- real_t * z
- real_t * y
- real_t * znew
- real_t * ynew
- real_t * temp1_dim_N
- uint32_t maxiter
- real_t eps

3.11.1 Detailed Description

Struct containing all necessary vectors and parameters for running FGM

Definition at line 20 of file fgm.h.

3.11.2 Field Documentation

3.11.2.1 real_t* Struct_FGM::c

Definition at line 23 of file fgm.h.

3.11.2.2 real_t Struct_FGM::eigH_max

Definition at line 36 of file fgm.h.

3.11.2.3 real_t Struct_FGM::eigH_min

Definition at line 37 of file fgm.h.

3.11.2.4 real_t Struct_FGM::eps

Definition at line 48 of file fgm.h.

3.11.2.5 uint32_t Struct_FGM::exitflag

Definition at line 31 of file fgm.h.

3.11.2.6 real_t Struct_FGM::fopt

Definition at line 30 of file fgm.h.

3.11.2.7 real_t* Struct_FGM::H

Definition at line 22 of file fgm.h.

3.11.2.8 real_t* Struct_FGM::lb

Definition at line 24 of file fgm.h.

3.11.2.9 boolean Struct_FGM::lb_is_inf

Definition at line 34 of file fgm.h.

3.11.2.10 uint32_t Struct_FGM::maxiter

Definition at line 47 of file fgm.h.

3.11.2.11 real_t* Struct_FGM::temp1_dim_N

Definition at line 44 of file fgm.h.

3.11.2.12 real_t* Struct_FGM::ub

Definition at line 25 of file fgm.h.

3.11.2.13 boolean Struct_FGM::ub_is_inf

Definition at line 35 of file fgm.h.

3.11.2.14 real_t* Struct_FGM::y

Definition at line 41 of file fgm.h.

3.11.2.15 real_t* Struct_FGM::ynew

Definition at line 43 of file fgm.h.

3.11.2.16 real_t* Struct_FGM::z

Definition at line 40 of file fgm.h.

3.11.2.17 real_t* Struct_FGM::z0

Definition at line 26 of file fgm.h.

3.11.2.18 real_t* Struct_FGM::znew

Definition at line 42 of file fgm.h.

3.11.2.19 real_t* Struct_FGM::zopt

Definition at line 29 of file fgm.h.

The documentation for this struct was generated from the following file:

- include/fgm.h

3.12 Struct_GDM Struct Reference

```
#include <gdm.h>
```

Data Fields

- real_t * H
- real_t * c
- real_t * lb
- real_t * ub
- real_t * z0
- real_t * zopt

- real_t fopt
- uint32_t exitflag
- boolean lb_is_inf
- boolean ub_is_inf
- real_t eigH_max
- real_t eigH_min
- real_t ∗ z
- real_t ∗ znew
- real_t ∗ temp1_dim_N
- uint32_t maxiter
- real_t eps

3.12.1 Detailed Description

Definition at line 15 of file gdm.h.

3.12.2 Field Documentation

3.12.2.1 real_t∗ Struct_GDM::c

Definition at line 18 of file gdm.h.

3.12.2.2 real_t Struct_GDM::eigH_max

Definition at line 31 of file gdm.h.

3.12.2.3 real_t Struct_GDM::eigH_min

Definition at line 32 of file gdm.h.

3.12.2.4 real_t Struct_GDM::eps

Definition at line 43 of file gdm.h.

3.12.2.5 uint32_t Struct_GDM::exitflag

Definition at line 26 of file gdm.h.

3.12.2.6 real_t Struct_GDM::fopt

Definition at line 25 of file gdm.h.

3.12.2.7 real_t∗ Struct_GDM::H

Definition at line 17 of file gdm.h.

3.12.2.8 real_t∗ Struct_GDM::lb

Definition at line 19 of file gdm.h.

3.12.2.9 boolean Struct_GDM::lb_is_inf

Definition at line 29 of file gdm.h.

3.12.2.10 uint32_t Struct_GDM::maxiter

Definition at line 42 of file gdm.h.

3.12.2.11 real_t* Struct_GDM::temp1_dim_N

Definition at line 39 of file gdm.h.

3.12.2.12 real_t* Struct_GDM::ub

Definition at line 20 of file gdm.h.

3.12.2.13 boolean Struct_GDM::ub_is_inf

Definition at line 30 of file gdm.h.

3.12.2.14 real_t* Struct_GDM::z

Definition at line 35 of file gdm.h.

3.12.2.15 real_t* Struct_GDM::z0

Definition at line 21 of file gdm.h.

3.12.2.16 real_t* Struct_GDM::znew

Definition at line 37 of file gdm.h.

3.12.2.17 real_t* Struct_GDM::zopt

Definition at line 24 of file gdm.h.

The documentation for this struct was generated from the following file:

- include/gdm.h

4 File Documentation

4.1 alm.c File Reference

```
#include "alm.h"
```

Functions

- int32_t ALM (struct Struct_ALM *s)

4.1.1 Function Documentation

4.1.1.1 int32_t ALM (struct Struct_ALM * *s*)

Definition at line 20 of file alm.c.

4.2 dfgm.c File Reference

```
#include "dfgm.h"
```

Functions

- int32_t DFGM (struct Struct_DFGM *s)

4.2.1 Function Documentation

4.2.1.1 int32_t DFGM (struct Struct_DFGM * *s*)

Definition at line 20 of file dfgm.c.

4.3 dgm.c File Reference

```
#include "dgm.h"
```

Functions

- int32_t DGM (struct Struct_DGM *s)

4.3.1 Function Documentation

4.3.1.1 int32_t DGM (struct Struct_DGM * *s*)

Definition at line 20 of file dgm.c.

4.4 falm.c File Reference

```
#include "falm.h"
```

Functions

- int32_t FALM (struct Struct_FALM *s)

4.4.1 Function Documentation

4.4.1.1 int32_t FALM (struct Struct_FALM * *s*)

Definition at line 20 of file falm.c.

4.5 fgm.c File Reference

```
#include "fgm.h"
```

Functions

- uint32_t FGM (struct Struct_FGM *s)
- void clean_up_FGM_C (struct Struct_FGM *s)

4.5.1 Function Documentation

4.5.1.1 void clean_up_FGM_C (struct Struct_FGM * s)

Definition at line 114 of file fgm.c.

4.5.1.2 uint32_t FGM (struct Struct_FGM * s)

Definition at line 12 of file fgm.c.

4.6 gdm.c File Reference

```
#include "gdm.h"
```

Functions

- uint32_t GDM (struct Struct_GDM *s)
- void clean_up_GDM_C (struct Struct_GDM *s)

4.6.1 Function Documentation

4.6.1.1 void clean_up_GDM_C (struct Struct_GDM * s)

Definition at line 93 of file gdm.c.

4.6.1.2 uint32_t GDM (struct Struct_GDM * s)

Definition at line 7 of file gdm.c.

4.7 general_functions.c File Reference

```
#include "general_functions.h"
```

Functions

- real_t * vector_alloc (uint32_t size)
- void free_pointer (real_t *pointer)
- void initArray (struct Array *a, uint32_t initialSize)
- void insertArray (struct Array *a, real_t element)
- void freeArray (struct Array *a)

4.7.1 Function Documentation

4.7.1.1 void free_pointer (real_t * pointer)

Definition at line 16 of file general_functions.c.

4.7.1.2 void freeArray (struct Array * a)

Definition at line 37 of file general_functions.c.

4.7.1.3 void initArray (struct Array * *a,* uint32_t *initialSize*)

Definition at line 23 of file general_functions.c.

4.7.1.4 void insertArray (struct Array * *a,* real_t *element*)

Definition at line 29 of file general_functions.c.

4.7.1.5 real_t* vector_alloc (uint32_t *size*)

Definition at line 10 of file general_functions.c.

4.8 include/alm.h File Reference

```
#include "head.h"
#include "math_functions.h"
#include "fgm.h"
#include "print.h"
```

Data Structures

- struct Struct_ALM

Functions

- int32_t ALM (struct Struct_ALM *s)

4.8.1 Detailed Description

Augmented Lagrangian Method

Definition in file alm.h.

4.8.2 Function Documentation

4.8.2.1 int32_t ALM (struct Struct_ALM * *s*)

Definition at line 20 of file alm.c.

4.9 include/dfgm.h File Reference

```
#include "head.h"
#include "math_functions.h"
#include "fgm.h"
#include "print.h"
```

Data Structures

- struct Struct_DFGM

Functions

- int32_t DFGM (struct Struct_DFGM *s)

4.9.1 Detailed Description

Dual Fast Gradient Method

Definition in file dfgm.h.

4.9.2 Function Documentation

4.9.2.1 int32_t DFGM (struct Struct_DFGM * s)

Definition at line 20 of file dfgm.c.

4.10 include/dgm.h File Reference

```
#include "head.h"
#include "math_functions.h"
#include "fgm.h"
#include "print.h"
```

Data Structures

- struct Struct_DGM

Functions

- int32_t DGM (struct Struct_DGM *s)

4.10.1 Detailed Description

Dual Gradient Method

Definition in file dgm.h.

4.10.2 Function Documentation

4.10.2.1 int32_t DGM (struct Struct_DGM * s)

Definition at line 20 of file dgm.c.

4.11 include/falm.h File Reference

```
#include "head.h"
#include "math_functions.h"
#include "fgm.h"
#include "print.h"
```

Data Structures

- struct Struct_FALM

Functions

- int32_t FALM (struct Struct_FALM *s)

4.11.1 Detailed Description

Fast Augmented Lagrangian Method

Definition in file falm.h.

4.11.2 Function Documentation

4.11.2.1 int32_t FALM (struct Struct_FALM * *s*)

Definition at line 20 of file falm.c.

4.12 include/fgm.h File Reference

```
#include "head.h"
#include <math.h>
#include "math_functions.h"
#include "print.h"
```

Data Structures

- struct Struct_FGM

Functions

- uint32_t FGM (struct Struct_FGM *s)
- void clean_up_FGM_C ()

4.12.1 Detailed Description

Fast Gradient Method

Definition in file fgm.h.

4.12.2 Function Documentation

4.12.2.1 void clean_up_FGM_C ( )

4.12.2.2 uint32_t FGM (struct Struct_FGM * *s*)

Definition at line 12 of file fgm.c.

4.13 include/gdm.h File Reference

```
#include "head.h"
#include "math_functions.h"
#include "general_functions.h"
```

Data Structures

- struct Struct_GDM

Functions

- uint32_t GDM ()
- void clean_up_GDM_C ()

4.13.1 Function Documentation

4.13.1.1 void clean_up_GDM_C ( )

4.13.1.2 uint32_t GDM ( )

4.14 include/general_functions.h File Reference

```
#include "head.h"
```

Functions

- real_t * vector_alloc ()
- void free_pointer (real_t *pointer)
- void initArray ()
- void insertArray ()
- void freeArray ()

4.14.1 Detailed Description

Contains some general functions that are common for most other files.

Definition in file general_functions.h.

4.14.2 Function Documentation

4.14.2.1 void free_pointer (real_t * *pointer*)

Definition at line 16 of file general_functions.c.

4.14.2.2 void freeArray ( )

4.14.2.3 void initArray ( )

4.14.2.4 void insertArray ( )

4.14.2.5 real_t* vector_alloc ( )

4.15 include/head.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <errno.h>
#include "typedefs.h"
#include "qp_structs.h"
```

Macros

- #define TRUE 1
- #define FALSE 0
- #define _DEBUG(fmt, args...)
- #define _DEBUG2(fmt, args...)
- #define YO printf("YOYO\n")
- #define ERROR(fmt) fprintf(stderr,"%s:%s:%d: "fmt, __FILE__, __FUNCTION__, __LINE__);
- #define _SDEBUG(fmt)

Variables

- uint32_t N
- uint32_t M
- uint32_t ALGORITHM

4.15.1 Detailed Description

Contains system libraries, mex libraries, global constants, global variables and some debugging macros.

Definition in file head.h.

4.15.2 Macro Definition Documentation

4.15.2.1 #define _DEBUG(*fmt, args...*)

Definition at line 41 of file head.h.

4.15.2.2 #define _DEBUG2(*fmt, args...*)

Definition at line 46 of file head.h.

4.15.2.3 #define _SDEBUG(*fmt*)

Definition at line 60 of file head.h.

4.15.2.4 #define ERROR(*fmt*) fprintf(stderr,"%s:%s:%d: "fmt, __FILE__, __FUNCTION__, __LINE__);

Definition at line 54 of file head.h.

4.15.2.5 #define FALSE 0

Definition at line 30 of file head.h.

4.15.2.6 #define TRUE 1

Definition at line 29 of file head.h.

4.15.2.7 #define YO printf("YOYO\n")

Definition at line 53 of file head.h.

4.15.3 Variable Documentation

4.15.3.1 uint32_t ALGORITHM

Definition at line 35 of file head.h.

4.15.3.2 uint32_t M

Dimension of the linear constraint matrix

Definition at line 34 of file head.h.

4.15.3.3 uint32_t N

Dimension of the Hessian matrix

Definition at line 33 of file head.h.

4.16 include/math_functions.h File Reference

```
#include "head.h"
```

Functions

- void mtx_vec_mul (const real_t *mtx, const real_t *v, real_t *res, const uint32_t rows, const uint32_t cols)
- void mtx_transpose (const real_t *mtx, real_t *mtx_t, const uint32_t rows, const uint32_t cols)
- void vector_min (const real_t *v1, const real_t *v2, real_t *res, const uint32_t length)
- void vector_max (const real_t *v1, const real_t *v2, real_t *res, const uint32_t length)
- void vector_sub (const real_t *v1, const real_t *v2, real_t *res, const uint32_t length)
- void vector_add (const real_t *v1, const real_t *v2, real_t *res, const uint32_t length)
- real_t vector_mul (const real_t *v1, const real_t *v2, const uint32_t length)
- void vector_scalar_mul (const real_t *v1, const real_t scalar, real_t *res, const uint32_t length)
- void vector_elements_to_zero (real_t *v, const uint32_t length)
- uint32_t vector_is_equal (const real_t *v1, const real_t *v2, const uint32_t length)
- void vector_copy (const real_t *v1, real_t *v2, const uint32_t length)
- void vector_max_with_zero (real_t *v, const uint32_t length)
- real_t vector_norm_2 (real_t *v, const uint32_t length)
- real_t abs_2 (const real_t a)
- real_t obj (const real_t *z, const real_t *H, const real_t *c, real_t *temp)

4.16.1 Detailed Description

Header for the internal math library. Contains basic vector and matrix operations. The library is tried to be as optimized as possible.

Definition in file math_functions.h.

4.16.2 Function Documentation

4.16.2.1 real_t abs_2 (const real_t *a*)

return the absolute value of the input

Definition at line 177 of file math_functions.c.

4.16.2.2 void mtx_transpose (const real_t * *mtx,* real_t * *mtx_t,* const uint32_t *rows,* const uint32_t *cols*)

Computes the transpose of the matrix: mtx_t = transpose(mtx)

Definition at line 52 of file math_functions.c.

4.16.2.3 void mtx_vec_mul (const real_t * *mtx,* const real_t * *v,* real_t * *res,* const uint32_t *rows,* const uint32_t *cols*)

matrix-vector multiplication: res = mtx * v

Definition at line 23 of file math_functions.c.

4.16.2.4 real_t obj (const real_t * *z,* const real_t * *H,* const real_t * *c,* real_t * *temp*)

compute the primal objective function

Definition at line 184 of file math_functions.c.

4.16.2.5 void vector_add (const real_t * *v1,* const real_t * *v2,* real_t * *res,* const uint32_t *length*)

vector addition res = v1 + v2

Definition at line 97 of file math_functions.c.

4.16.2.6 void vector_copy (const real_t * *v1,* real_t * *v2,* const uint32_t *length*)

copy a vector element by element: v2 = v1

Definition at line 141 of file math_functions.c.

4.16.2.7 void vector_elements_to_zero (real_t * *v,* const uint32_t *length*)

set all elements in vector v to zero

Definition at line 123 of file math_functions.c.

4.16.2.8 uint32_t vector_is_equal (const real_t * *v1,* const real_t * *v2,* const uint32_t *length*)

returns true if all elements in the two vectors are equal

Definition at line 131 of file math_functions.c.

4.16.2.9 void vector_max (const real_t * *v1,* const real_t * *v2,* real_t * *res,* const uint32_t *length*)

compare two vectors element by element, and returns the largest element of the two: res = max(v1,v2)

Definition at line 78 of file math_functions.c.

4.16.2.10 void vector_max_with_zero (real_t * *v,* const uint32_t *length*)

project a vector on all positive numbers: v = max(v,0.0)

Definition at line 149 of file math_functions.c.

4.16.2.11 void vector_min (const real_t * *v1,* const real_t * *v2,* real_t * *res,* const uint32_t *length*)

compare two vectors element by element, and returns the smallest element of the two: res = min(v1,v2)

Definition at line 67 of file math_functions.c.

4.16.2.12 real_t vector_mul (const real_t * v1, const real_t * v2, const uint32_t length)

vector multiplication: res = v1 * v2

Definition at line 105 of file math_functions.c.

4.16.2.13 real_t vector_norm_2 (real_t * v, const uint32_t length)

compute the euclidean norm of the vector v

Definition at line 158 of file math_functions.c.

4.16.2.14 void vector_scalar_mul (const real_t * v1, const real_t scalar, real_t * res, const uint32_t length)

Vector times a scalar: res = v1 * scalar

Definition at line 115 of file math_functions.c.

4.16.2.15 void vector_sub (const real_t * v1, const real_t * v2, real_t * res, const uint32_t length)

vector subtract: res = v1 - v2

Definition at line 89 of file math_functions.c.

4.17 include/print.h File Reference

```
#include "head.h"
#include "gdm.h"
#include "fgm.h"
#include "dgm.h"
#include "dfgm.h"
```

Functions

- void print_vector ()
- void print_matrix ()
- void print_Problem ()
- void print_Options ()
- void print_Info ()
- void print_Result ()
- void print_problem_FGM ()
- void print_result_FGM ()
- void print_result_GDM ()

4.17.1 Function Documentation

4.17.1.1 void print_Info ( )

4.17.1.2 void print_matrix ( )

4.17.1.3 void print_Options ( )

4.17.1.4 void print_Problem ( )

4.17.1.5 void print_problem_FGM ( )

4.17.1.6 void print_Result ( )

4.17.1.7 void print_result_FGM ( )

4.17.1.8 void print_result_GDM ( )

4.17.1.9 void print_vector ( )

4.18 include/qp_structs.h File Reference

```
#include "typedefs.h"
```

Data Structures

- struct Problem
- struct Options
- struct Info
- struct Output
- struct Result
- struct Array

4.18.1 Detailed Description

Contains the declaration of the structs that are common for the algorithms

Definition in file qp_structs.h.

4.19 include/typedefs.h File Reference

Typedefs

- typedef char char_t
- typedef signed char int8_t
- typedef signed short int16_t
- typedef signed int int32_t
- typedef unsigned char uint8_t
- typedef unsigned short uint16_t
- typedef unsigned int uint32_t
- typedef float float32_t
- typedef double float64_t
- typedef uint32_t boolean
- typedef float64_t real_t

4.19.1 Typedef Documentation

4.19.1.1 typedef uint32_t boolean

Definition at line 24 of file typedefs.h.

4.19.1.2 typedef char char_t

Definition at line 11 of file typedefs.h.

4.19.1.3 typedef float float32_t

Definition at line 21 of file typedefs.h.

4.19.1.4 typedef double float64_t

Definition at line 22 of file typedefs.h.

4.19.1.5 typedef signed short int16_t

Definition at line 14 of file typedefs.h.

4.19.1.6 typedef signed int int32_t

Definition at line 15 of file typedefs.h.

4.19.1.7 typedef signed char int8_t

Definition at line 13 of file typedefs.h.

4.19.1.8 typedef float64_t real_t

Definition at line 25 of file typedefs.h.

4.19.1.9 typedef unsigned short uint16_t

Definition at line 18 of file typedefs.h.

4.19.1.10 typedef unsigned int uint32_t

Definition at line 19 of file typedefs.h.

4.19.1.11 typedef unsigned char uint8_t

Definition at line 17 of file typedefs.h.

4.20 main.c File Reference

```
#include "head.h"
#include "gdm.h"
#include "fgm.h"
#include "dgm.h"
#include "dfgm.h"
#include "alm.h"
#include "falm.h"
#include "print.h"
```

Macros

- #define H_IN prhs[0]
- #define C_IN prhs[1]
- #define A_IN prhs[2]
- #define B_IN prhs[3]
- #define LB_HAT_IN prhs[4]
- #define UB_HAT_IN prhs[5]
- #define LB_IN prhs[6]
- #define UB_IN prhs[7]
- #define Z0_IN prhs[8]
- #define OPT_IN prhs[9]
- #define INFO_IN prhs[10]
- #define H_HAT_IN prhs[11]

- #define A2_IN prhs[12]
- #define RHO_AT_B_IN prhs[13]
- #define ZOPT_OUT plhs[0]
- #define FOPT_OUT plhs[1]
- #define EXITFLAG_OUT plhs[2]
- #define OUTPUT_STRUCT_OUT plhs[3]
- #define LAMBDA1_OUT plhs[4]
- #define LAMBDA2_OUT plhs[5]
- #define NUM_FIELDS 11
- #define NAME_LENGT 30
- #define ITERATION 0
- #define ITERATION_INNER_TOT 1
- #define TIME 2
- #define TIME_TOT_INNER 3
- #define FLAG_LAST_SATISFIED 4
- #define NITER_FEASIBLE_DS 5
- #define NITER_FEASIBLE_PF 6
- #define EXITFLAG_INNER 7
- #define NUM_EXCEEDED_MAX_NITER_INNER 8
- #define DS_VECTOR 9
- #define PF_VECTOR 10

Functions

- void mexFunction (int nlhs, mxArray ∗plhs[], int nrhs, const mxArray ∗prhs[])

4.20.1 Detailed Description

Main file which takes input from matlab and construct data in c-code format. This mainfile is called from the matlab function duquad.m. After converting the input, it runs the spesified algorithm, i.e one of the following: DGM, DFGM, ALM or FALM. When getting the result from the algorithm, the data is converted back to matlab format.

Definition in file main.c.

4.20.2 Macro Definition Documentation

4.20.2.1 #define A2_IN prhs[12]

Definition at line 33 of file main.c.

4.20.2.2 #define A_IN prhs[2]

Definition at line 21 of file main.c.

4.20.2.3 #define B_IN prhs[3]

Definition at line 22 of file main.c.

4.20.2.4 #define C_IN prhs[1]

Definition at line 20 of file main.c.

4.20.2.5 #define DS_VECTOR 9

4.20.2.6 #define EXITFLAG_INNER 7

4.20.2.7 #define EXITFLAG_OUT plhs[2]

Definition at line 39 of file main.c.

4.20.2.8 #define FLAG_LAST_SATISFIED 4

4.20.2.9 #define FOPT_OUT plhs[1]

Definition at line 38 of file main.c.

4.20.2.10 #define H_HAT_IN prhs[11]

Definition at line 32 of file main.c.

4.20.2.11 #define H_IN prhs[0]

Definition at line 19 of file main.c.

4.20.2.12 #define INFO_IN prhs[10]

Definition at line 29 of file main.c.

4.20.2.13 #define ITERATION 0

4.20.2.14 #define ITERATION_INNER_TOT 1

4.20.2.15 #define LAMBDA1_OUT plhs[4]

Definition at line 41 of file main.c.

4.20.2.16 #define LAMBDA2_OUT plhs[5]

Definition at line 42 of file main.c.

4.20.2.17 #define LB_HAT_IN prhs[4]

Definition at line 23 of file main.c.

4.20.2.18 #define LB_IN prhs[6]

Definition at line 25 of file main.c.

4.20.2.19 #define NAME_LENGT 30

4.20.2.20 #define NITER_FEASIBLE_DS 5

4.20.2.21 #define NITER_FEASIBLE_PF 6

4.20.2.22 #define NUM_EXCEEDED_MAX_NITER_INNER 8

4.20.2.23 #define NUM_FIELDS 11

4.20.2.24 #define OPT_IN prhs[9]

Definition at line 28 of file main.c.

4.20.2.25 #define OUTPUT_STRUCT_OUT plhs[3]

Definition at line 40 of file main.c.

4.20.2.26 #define PF_VECTOR 10

4.20.2.27 #define RHO_AT_B_IN prhs[13]

Definition at line 34 of file main.c.

4.20.2.28 #define TIME 2

4.20.2.29 #define TIME_TOT_INNER 3

4.20.2.30 #define UB_HAT_IN prhs[5]

Definition at line 24 of file main.c.

4.20.2.31 #define UB_IN prhs[7]

Definition at line 26 of file main.c.

4.20.2.32 #define Z0_IN prhs[8]

Definition at line 27 of file main.c.

4.20.2.33 #define ZOPT_OUT plhs[0]

Definition at line 37 of file main.c.

4.20.3 Function Documentation

4.20.3.1 void mexFunction (int *nlhs*, mxArray * *plhs[]*, int *nrhs*, const mxArray * *prhs[]*)

Definition at line 74 of file main.c.

4.21 math_functions.c File Reference

```
#include "math_functions.h"
```

Functions

- void mtx_vec_mul (const real_t *mtx, const real_t *v, real_t *res, const uint32_t rows, const uint32_t cols)
- void mtx_transpose (const real_t *mtx, real_t *mtx_t, const uint32_t rows, const uint32_t cols)
- void vector_min (const real_t *v1, const real_t *v2, real_t *res, const uint32_t length)
- void vector_max (const real_t *v1, const real_t *v2, real_t *res, const uint32_t length)
- void vector_sub (const real_t *v1, const real_t *v2, real_t *res, const uint32_t length)
- void vector_add (const real_t *v1, const real_t *v2, real_t *res, const uint32_t length)
- real_t vector_mul (const real_t *v1, const real_t *v2, const uint32_t length)
- void vector_scalar_mul (const real_t *v1, const real_t scalar, real_t *res, const uint32_t length)
- void vector_elements_to_zero (real_t *v, const uint32_t length)
- uint32_t vector_is_equal (const real_t *v1, const real_t *v2, const uint32_t length)
- void vector_copy (const real_t *v1, real_t *v2, const uint32_t length)
- void vector_max_with_zero (real_t *v, const uint32_t length)
- real_t vector_norm_2 (real_t *v, const uint32_t length)
- real_t abs_2 (const real_t a)
- real_t obj (const real_t *z, const real_t *H, const real_t *c, real_t *temp)

4.21.1 Function Documentation

4.21.1.1 real_t abs_2 (const real_t *a*)

return the absolute value of the input

Definition at line 177 of file math_functions.c.

4.21.1.2 void mtx_transpose (const real_t * *mtx,* real_t * *mtx_t,* const uint32_t *rows,* const uint32_t *cols*)

Computes the transpose of the matrix: mtx_t = transpose(mtx)

Definition at line 52 of file math_functions.c.

4.21.1.3 void mtx_vec_mul (const real_t * *mtx,* const real_t * *v,* real_t * *res,* const uint32_t *rows,* const uint32_t *cols*)

matrix-vector multiplication: res = mtx * v

Definition at line 23 of file math_functions.c.

4.21.1.4 real_t obj (const real_t * *z,* const real_t * *H,* const real_t * *c,* real_t * *temp*)

compute the primal objective function

Definition at line 184 of file math_functions.c.

4.21.1.5 void vector_add (const real_t * *v1,* const real_t * *v2,* real_t * *res,* const uint32_t *length*)

vector addition res = v1 + v2

Definition at line 97 of file math_functions.c.

4.21.1.6 void vector_copy (const real_t * *v1,* real_t * *v2,* const uint32_t *length*)

copy a vector element by element: v2 = v1

Definition at line 141 of file math_functions.c.

4.21.1.7 void vector_elements_to_zero (real_t * *v,* const uint32_t *length*)

set all elements in vector v to zero

Definition at line 123 of file math_functions.c.

4.21.1.8 uint32_t vector_is_equal (const real_t * *v1,* const real_t * *v2,* const uint32_t *length*)

returns true if all elements in the two vectors are equal

Definition at line 131 of file math_functions.c.

4.21.1.9 void vector_max (const real_t * *v1,* const real_t * *v2,* real_t * *res,* const uint32_t *length*)

compare two vectors element by element, and returns the largest element of the two: res = max(v1,v2)

Definition at line 78 of file math_functions.c.

4.21.1.10 void vector_max_with_zero (real_t * *v,* const uint32_t *length*)

project a vector on all positive numbers: v = max(v,0.0)

Definition at line 149 of file math_functions.c.

4.21.1.11 void vector_min (const real_t * *v1,* const real_t * *v2,* real_t * *res,* const uint32_t *length*)

compare two vectors element by element, and returns the smallest element of the two: res = min(v1,v2)

Definition at line 67 of file math_functions.c.

4.21.1.12 real_t vector_mul (const real_t * v1, const real_t * v2, const uint32_t length)

vector multiplication: res = v1 * v2

Definition at line 105 of file math_functions.c.

4.21.1.13 real_t vector_norm_2 (real_t * v, const uint32_t length)

compute the euclidean norm of the vector v

Definition at line 158 of file math_functions.c.

4.21.1.14 void vector_scalar_mul (const real_t * v1, const real_t scalar, real_t * res, const uint32_t length)

Vector times a scalar: res = v1 * scalar

Definition at line 115 of file math_functions.c.

4.21.1.15 void vector_sub (const real_t * v1, const real_t * v2, real_t * res, const uint32_t length)

vector subtract: res = v1 - v2

Definition at line 89 of file math_functions.c.

4.22 print.c File Reference

```
#include "print.h"
```

Functions

- void print_vector (double *v, int length)
- void print_matrix (const real_t *mtx, const uint32_t rows, const uint32_t cols)
- void print_Problem (struct Problem *s)
- void print_Options (struct Options *opt)
- void print_Info (struct Info *info)
- void print_Result (struct Result *s)
- void print_problem_FGM (struct Struct_FGM *s)
- void print_result_FGM (struct Struct_FGM *s, int niter)
- void print_result_GDM (struct Struct_GDM *s, int niter)

4.22.1 Function Documentation

4.22.1.1 void print_Info (struct Info * info)

Definition at line 64 of file print.c.

4.22.1.2 void print_matrix (const real_t * mtx, const uint32_t rows, const uint32_t cols)

Definition at line 20 of file print.c.

4.22.1.3 void print_Options (struct Options * opt)

Definition at line 52 of file print.c.

4.22.1.4 void print_Problem (struct Problem * s)

Definition at line 35 of file print.c.

4.22.1.5 void print_problem_FGM (struct Struct_FGM * *s*)

Definition at line 102 of file print.c.

4.22.1.6 void print_Result (struct Result * *s*)

Definition at line 79 of file print.c.

4.22.1.7 void print_result_FGM (struct Struct_FGM * *s*, int *niter*)

Definition at line 119 of file print.c.

4.22.1.8 void print_result_GDM (struct Struct_GDM * *s*, int *niter*)

Definition at line 129 of file print.c.

4.22.1.9 void print_vector (double * *v*, int *length*)

Definition at line 9 of file print.c.

Index