Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

ModFEM - solvers of linear equations

ModFEM – solvers of linear equations

- two interfaces:
  - mkb
    - for iterative as well as direct solvers
  - direct
    - one function: sir_direct_solve_lin_sys
- choice in problem definition file:
  - linear_solver_type
    - direct: -1
      - (now 0, but direct solver interface is considered obsolete and default 0 value is going to direct solvers through mkb interface with -1 assigned to old direct solver interface)
    - mkb: >=0
      - was >0 but now includes 0 as direct solver)

ModFEM – solvers of linear equations

- interfacing ModFEM with linear solvers
  - old idea:
    - solvers are different (from each other)
    - they require different solver interfaces in ModFEM (sid_... packages)
      - sid_.... routines know ModFEM
      - sid_routines not necessarily know original solver interface and data structures
    - still solvers may require adapters (lsd_... packages)
      - lsd_... routines know solver interface and data structures
      - lsd_... routines implement different interfaces, depending on sid_... packages
      - sid_... routines know the interface of the corresponding lsd_... package (and the place of lsh_....h files – they are not in ModFEM include directory)

ModFEM – solvers of linear equations

- sid_lapack (OBSOLETE)
  - “oldie but goldie”, “mother of all solver modules”, still works!!!??? (at least compiles...)
  - uses old DOF structures
  - does not support:
    - coloring and OpenMP assembly
    - external renumbering (serious problem)
  - is devilishly slow...
  - but has some debug prints...

ModFEM – solvers of linear equations

- sid_pardiso (OBSOLETE)
  - uses old DOF structures and **`std::vector<std::map<int, double> >`**
  - recreates system matrix for each solve call...
  - does not support:
    - coloring and OpenMP assembly
    - external renumbering (not a problem it has internal renumbering)
- PARDISO is also available through MKB interface, where:
  - new DOF structures are used
  - coloring and OpenMP assembly is supported

ModFEM – solvers of linear equations

- interfacing ModFEM with linear solvers
  - new idea (after several years of practice):
    - solvers are different (from each other)
      - but not that different to require different solver interfaces
      - it turned out that all existing sid_... packages used sid_mkb data structures and basic logic of operation
    - hence: there is now one sid_mkb solver interface package
      - sid_mkb implements include/sih_intf.h interface
      - sid_mkb requires packages implementing sid_mkb/lsh_mkb_intf.h interface
      - sid_mkb routines do not know the interface and data structures of the final solver implementation
      - sid_mkb functions interact only with problem modules (so that each problem module can adapt to the solver) and use several (three as for now) general purpose utilities

ModFEM – solvers of linear equations

- sid_mkb
  - universal interface for direct and iterative solvers (mkb is now a proper name...)
  - part of ModFEM as its internal module
  - supports SOLVE and RESOLVE
  - supports reuse of matrix data structure (with possible symbolic factorization retained)
  - supports OpenMP and OpenCL assembly with assembly tables
  - supports arbitrary renumbering
  - supports MPI distributed memory parallelism
  - supports geometric multigrid (also with MPI)
  - supports several solver interface instances for different (coupled) problems

ModFEM – solvers of linear equations

- sid_mkb functions:
  - sir_module_introduce – to differentiate sid_... modules (if problem modules want to)
  - sir_init – stage 1: to initialize a solver instance and read its configuration file (if present)
  - sir_create – stage 2: to create data structures related to a particular system of equations
  - sir_solve – stage 3: to fill data structures for a given solve (SM and LV for SOLVE, LV only for RESOLVE) and then solve (creating preconditioner for iterative solvers) – hence several sub-stages
  - sir_free – stage 4: to free data structures related to a particular system of equations
  - sir_destroy – stage 5: to remove a solver instance
  - sir_solve_lin_sys – all 5 stages in one call

ModFEM – solvers of linear equations

- sid_mkb required FEM interface:
  - pdr_get_list_ent
    - provides solver with the lists of types and identifiers for integration and DOF entities
  - pdr_get_list_ent_coarse
    - given lists for the fine mesh, it coarsens the mesh geometrically and returns the lists for the coarse mesh
  - pdr_comp_stiff_mat
    - supplies solver with element and face stiffness matrices and load vectors, together with lists of DOF entities (unconstrained only!) - the only source for connectivity information given to the solver!

ModFEM – solvers of linear equations

- sid_mkb required FEM interface:
  - pdr_create_assemble_stiff_mat
    - to create stiffness matrices for integration entities on the provided list and assemble them to the solver data structure
    - should support assembling with provided tables
    - should support OpenMP and/or OpenCL assembly
  - pdr_read_sol_dofs
    - to read values of DOFs (for initial condition)
  - pdr_write_sol_dofs
    - to write the final solution to the FEM data structure

ModFEM – solvers of linear equations

- sid_mkb required FEM interface for problem independent utilities:
  - utr_renumber
    - to renumber DOFs
    - currently only Reverse Cuthill-McKee
  - utr_color_int_ent_for_assembly
    - to make multi-threaded assembly possible without critical sections
  - utr_io_write_img_to_pbm
    - to visualize the structure of the global stiffness matrix

ModFEM – solvers of linear equations

- additional sid_mkb required FEM interface for PARALLEL (MPI) execution (lsd_mkb_core solver only):
  - pdr_create_exchange_tables
    - to create the data structure for exchange of DOF values during solver iterations
  - pdr_exchange_dofs
    - to exchange DOF values in a provided vector
    - must be able to link positions in the input global vector to the types and identifiers of DOF entities in the mesh
  - pdr_vec_norm
    - to compute the standard L2 norm for the global vector, given the subdomain part of it as input
  - pdr_sc_prod
    - the same for the scalar product of two global vectors

ModFEM – solvers of linear equations

- additional sid_mkb required FEM interface for PARALLEL (MPI) execution (lsd_mkb_core multigrid solver for DG approximation only):
  - pdr_dof_ent_sons
    - to return a list of sons of a DOF entity (for DG, elements are the only DOF entities)
  - pdr_get_ent_pdeg
    - to return the degree of approximation for a DOF entity
  - pdr_L2_proj_sol
    - to project the solution from an element to its sons and vice versa

ModFEM – solvers of linear equations

- sid_mkb
  - for each problem (subproblem in coupled problems) a separate solver interface instance can exist!
    - generally there is ono-to-one correspondence problem-solver interface instance-linear solver instance
    - for geometric multigrid for one problem there is one solver interface instance with several mesh levels and one linear solver instance with possibly different solver algorithms for each level
    - for ns_supg extensions for one problem and one solver interface instance there is one linear solver instance but several subproblems each with possibly many levels and a separate solver algorithm for each level but...
    - fortunately there is no geometric multigrid for ns_supg_ext yet
  - each solver module holds a table with pointers to particular solver interface instances (of type sit_solvers)
  - each solver module can handle up to 10 solver instances (magic number... - SIC_MAX_NUM_SOLV = 10)

ModFEM – solvers of linear equations

- sid_mkb
  - sid_mkb functions call functions declared in lsh_mkb_intf.h interface
  - these functions should be implemented by adapter modules for possibly external linear solvers
  - the first lsh_mkb_intf.h function initializes the linear solver instance and passes all (!) parameters specified in problem input file
  - initialization function has, as one of its argument, the name of the configuration file for a particular solver, where arbitrary number of additional parameters can be specified
    - in lsd_mkb_core implementation parameters in solver configuration file override parameters in problem configuration file
  - most of the functions have two arguments as first on their lists:
    - Solver_id (for all the functions except the initialization function)
    - Level_id
  - it is assumed that the main linear solver function (lsr_mkb_solve) concerns the finest level

ModFEM – solvers of linear equations

- interfacing ModFEM with linear solvers (cont.)
  - new idea (after several years of practice):
    - there is one sid_mkb solver interface module
    - there is one lsd_mkb super-adapter package
    - there are many solver adapters and implementations (lsd_mkb/lsd_... packages)
    - all solver adapters and implementations use the same set of supporting linear algebra modules (lsd_mkb/lad_... packages)
      - for direct solvers linear algebra modules provide storage for system matrices
    - as with many incrementally evolving codes, the implementation is not clean – mixture of compile time polymorphism and good old if..elsif..

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package
  - creates instances of linear solvers
  - creates matrix data structures in selected formats
  - calls linear solvers:
    - direct - one implementation selected at compile time from:
      - SuperLU
      - PARDISO
      - ViennaCL
    - iterative - one implementation selected at compile time from:
      - mkb_core
      - amg (?)
      - ViennaCL (?)

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package functions:
  - declared in lsd_mkb/lsh_mkb_intf.h
  - three types of functions (for all solvers except special solver for ns_supg where specific handling of matrices takes places and all functions are solver functions):
    - solver only functions – do not concern SM and LV and its storage module (lad_...)
    - storage only functions – do not concern solver implementation (algorithms)
    - solver and storage functions – combine solver algorithms, SM and LV
      - usually implemented as a call to solver function with ID of SM and LV as arguments

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package functions:
  - lsr_mkb_init (corresponds to sir_init)
    - to create a new solver instance, read its control parameters and initialize its data structure
  - lsr_mkb_create_matrix (called by sir_create, together with lsr_mkb_create_precon – for iterative solvers)
    - to allocate space for a global system matrix
  - lsr_mkb_clear_matrix (part of sir_solve)
    - to initialize (clear) data structure of system matrix
  - lsr_mkb_free_matrix (corresponds to sir_free)
    - to free matrix data structure (and related structures, like preconditioner etc.)
  - lsr_mkb_destroy (corresponds to sir_destroy)
    - to destroy a particular instance of the solver

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package functions:
  - assembly stage – all functions for sir_solve
  - lsr_mkb_fill_assembly_table_int_ent
    - to create a part of the global assembly table related to one integration entity, for which lists of DOF blocks (their global positions) are provided
  - lsr_mkb_assemble_local_stiff_mat_with_table
    - to assemble entries to the global stiffness matrix and the global load vector using the provided local stiffness matrix, load vector and the proper part of the global assembly table
  - lsr_mkb_assemble_local_sm
    - to assemble entries to the global stiffness matrix and the global load vector using the provided local stiffness matrix and load vector

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package functions (for preconditioned Kryłow space solvers only):
  - lsr_mkb_create_precon
    - called by sir_create
    - to create preconditioner data structures
  - lsr_mkb_fill_precon
    - called by sir_solve
    - to prepare preconditioner by factorizing the stiffness matrix, either only diagonal blocks or (block) ILU(k)
  - lsr_get_pdeg_coarse (for geometric multigrid solver only)
    - to get enforced pdeg for the coarse mesh

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package functions:
  - lsr_mkb_solve
    - to solve a system of equations, given previously constructed system matrix, possibly preconditioner, etc.
    - the interface is designed for iterative solvers, but...
    - implementation in super-adapter module can call internally lsr_mkb_solve_direct function with the interface designed for direct solvers
      - lsr_mkb_solve_direct uses a matrix stored in lad_... module
      - if the direct solver selected has different storage format than lad_... module, special functions for rewriting system matrix must be used

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package – functions called:
  - solver management
    - now just separate calls for direct and iterative solvers
  - lsr_mkb_init (choice depends on Solver_type):
    - lsr_mkb_direct_init
    - lsr_mkb_core_init
  - lsr_mkb_solve (choice depends on Solver_type):
    - lsr_mkb_direct_solve
    - lsr_mkb_core_solve
    - lsr_ns_supg_ext_solve (possibly moved to lsr_mkb_core_solve)
  - lsr_mkb_destroy (choice depends on Solver_type):
    - lsr_mkb_direct_destroy
    - lsr_mkb_core_destroy

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package – functions called:
  - matrix management (solver algorithm independent – except for ns_supg solver and lsr_mkb_free_matrix...)
  - lsr_mkb_create_matrix
    - lar_allocate_SM_and_LV
    - lsr_ns_supg_ext_create_matrix (separate path for ns_supg solver)
  - lsr_mkb_clear_matrix
    - lar_initialize_SM_and_LV
    - lsr_ns_supg_ext_clear_matrix (separate path for ns_supg solver)
  - lsr_mkb_free_matrix
    - lar_free_SM_and_LV
    - lsr_ns_supg_ext_free_matrix (separate path for ns_supg solver)
    - lsr_mkb_core_destroy_precon (for iterative solvers only)

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package – functions called:
  - assembly stage (all functions for sir_solve)
    - fast path – assembly tables creation and use
    - old, generic, universal interface without assembly tables
  - lsr_mkb_fill_assembly_table_int_ent (fast path)
    - lar_fill_assembly_table_int_ent
  - lsr_mkb_assemble_local_stiff_mat_with_table (fast path)
    - lar_assemble_SM_and_LV_with_table
  - lsr_mkb_assemble_local_sm (old, generic, universal interface)
    - lar_assemble_SM_and_LV
    - lsr_ns_supg_ext_assemble_local_sm (separate path for ns_supg solver)

ModFEM – solvers of linear equations

- lsd_mkb super-adapter package – functions called:
  - preconditioner management (for iterative solvers only)
  - lsr_mkb_create_precon
    - lsr_mkb_core_create_precon
  - lsr_mkb_fill_precon
    - lsr_mkb_core_fill_precon
  - multigrid utility
  - lsr_mkb_get_pdeg_coarse
    - lsr_mkb_core_get_pdeg_coarse

ModFEM – solvers of linear equations

- direct solvers (algorithms):
  - two interfaces possible:
    - “direct”, i.e. sid_...
    - through mkb
  - implementations in external libraries:
    - SuperLU – new default (through mkb interface)
    - PARDISO – if MKL available (directly or through mkb interface)
    - LAPACK – no one knows why... (was necessary before SuperLU and without MKL)
    - ViennaCL – if library available (through sid_Viennacl_crs or through sid_mkb ? - probably should be updated...)

ModFEM – solvers of linear equations

- direct solvers (algorithms):
  - currently only OpenMP implementations (no MPI)
  - for direct solver interface:
    - self contained implementations in sid_... directories of sir_direct_solve_lin_sys routine from include/sih_intf.h interface
      - one significant argument: Filename – the name of configuration file (if empty defaults are used)
      - solver parameters possible to set only using this file
  - for mkb interface:
    - solver is:
      - initialized (using possibly the name of configuration file)
      - called for solution with SM and LV ID as parameter
      - destroyed.
    - SM and LV handling is separated from solver

ModFEM – solvers of linear equations

- direct solvers (algorithms) through mkb interface:
  - an array of solver instances with assigned IDs
  - lsr_mkb_direct_init(Solver_id, Filename, Monitor);
    - Filename – configuration file
    - Monitor – always: monitoring level
  - lsr_mkb_direct_solve(Solver_id, Comp_type, SM_and_LV_id, Ndof, X, B, Monitor);
    - Comp_type – LSC_SOLVE or LSC_RESOLVE
    - SM_and_LV_id – ID of SM and LV in storage module
    - Ndof – total number of DOFs
    - X – solution – output
    - B – RHS (LV) vector
  - lsr_mkb_direct_destroy(Solver_id);

ModFEM – solvers of linear equations

- iterative solvers:
  - implementations:
    - mkb_core (in lsd_mkb/lsd_mkb_core)
      - original block iterative solver
      - now: set of algorithms for multigrid preconditioned GMRES (and possibly other Kryłow space solvers)
    - amg – through sid_amg or through sid_mkb ?
      - can it be combined with mkb_core?
    - ViennaCL – if library available (through sid_viennacl_crs or through sid_mkb ? - probably should be updated...)
    - ns_supg_ext – special extensions for ns_supg problem
      - the original system is split into four parts with special algorithms for preconditioning

ModFEM – solvers of linear equations

- mkb_core iterative solver implementation:
  - many possible algorithms:
    - standard iterative solvers (block-Jacobi, block-GS)
    - geometric multigrid with different smoothers
      - block-Jacobi, block-GS
      - ILU
    - GMRES with preconditioning:
      - block-Jacobi, block-GS
      - ILU
      - geometric multigrid with different smoothers:
        - block-Jacobi, block-GS
        - ILU
      - amg ? - with different smoothers?

ModFEM – solvers of linear equations

- mkb_core iterative solver implementation:
  - interface and realization:
    - lsr_mkb_core_init
      - initiate solver instance and read control parameters from configuration file
    - lsr_mkb_core_create_precon
      - create preconditioner data structures for a given SM structure and a set of control parameters
    - lsr_mkb_core_fill_precon
      - fill preconditioner data structures for a given SM
    - lsr_mkb_core_solve
      - solve for a given SM and LV
    - lsr_mkb_core_destroy_precon
    - lsr_mkb_core_destroy
      - delete solver instance

ModFEM – solvers of linear equations

- all solver implementations interfacing through mkb, use a set of linear algebra supporting routines
- the routines are related to:
  - storage management for SM and LV
  - operations required by iterative solvers
- the interface is contained in lsd_mkb/lah_intf.h
- there is one implementation of interface in lsd_mkb/las_intf.c
  - routines in las_intf.c call implementations for different storage types contained in different lad_... directories
  - las_intf.c has to know all implementations for all storage types
    - not nice... - no compile time (static) polymorphism

ModFEM – solvers of linear equations

- linear algebra supporting routines
  - there are currently four storage options:
    - CRS
      - no blocks in SM
      - optimal for scalar problems
    - BCRS
      - constant size blocks in SM
      - optimal for vector problems
    - block
      - possible variable size of blocks in SM
      - standard for DG problems (has special preconditioners)
    - CRS_generic
      - possible variable size of blocks in SM
      - seem to be the most universal

ModFEM – solvers of linear equations

- linear algebra supporting routines
  - the selection of storage is based on parameter: storage_type
    - this parameter is passed from sir_create
      - there is an option for las_intf.c to decide on storage type, based e.g. on the existence and the size of blocks in SM
  - all implementations are always compiled with ModFEM
    - in coupled problems we can use different storage types for different sub-problems
  - the names of implementation routines are obtained by adding _crs or _bcrs or _block or _crs_generic at the end of las_intf.c routines

ModFEM – solvers of linear equations

- linear algebra supporting routines
  - storage management routines
    - called from lsd_mkb_intf.c:
      - lar_allocate_SM_and_LV
      - lar_initialize_SM_and_LV
      - lar_fill_assembly_table_int_ent
      - lar_assemble_SM_and_LV_with_table
      - lar_assemble_SM_and_LV
        - generic – no assembly tables (old style)
        - the only for block storage
      - lar_free_SM_and_LV

ModFEM – solvers of linear equations

- linear algebra supporting routines
  - preconditioner management routines
    - for iterative solvers only (currently mkb_core)
    - routines must know the storage requirements for particular preconditioner algorithms:
      - lar_allocate_preconditioner
        - » to allocate space for preconditioner
      - lar_fill_preconditioner
        - » to fill preconditioner data structures
        - » the routine must sometimes execute the actual algorithm (e.g. for ILU or overlapping subdomains preconditioning)
      - lar_free_preconditioner
        - » to free space of preconditioner structure

ModFEM – solvers of linear equations

- linear algebra supporting routines
  - iterative solver supporting routines:
    - lar_compute_residual -
      - to compute the residual of the not preconditioned system of equations, v = (b - Ax)
    - lar_compute_preconditioned_residual
      - to compute the residual of the preconditioned system of equations, v = M^-1 * (b - Ax), where M^-1 corresponds directly to the stored preconditioner matrix
    - lar_perform_BJ_or_GS_iterations
      - to perform one iteration of block Gauss-Seidel or block Jacobi algorithm: v_out = v_in + M^-1 * (b - A * v_in)
    - lar_perform_rhsub
      - to perform forward reduction and back-substitution for ILU preconditioning