Interfaces Between the Optimisation Shell INVERSE and Simulation Programmes

(FOR VERSION 3.11)

Igor Grešovnik

Ljubljana, 27 September, 2005

Contents:

12. SHELL-SIMULATION INTERFACES

12.1Elfen Interface

Elfen is a general purpose finite element programme for solution of thermo-mechanical problems in solid mechanics. It is capable of solving nonlinear problems involving large strains and deformations, different material models, thermo-mechanical coupling and contact phenomena. It is therefore convenient for simulation of a large range of forming processes or product behavior in operating conditions.

At the current stage, a file interface between the shell Inverse and Elfen is available. The interfacing utilities include reading individual results or result fields from Elfen's output files, and replacing fields in the Elfen's input files. It is usually much more comfortable to use these interfacing utilities than to use the general file interface.

A set of expression evaluator's functions that read individual results from the analysis output file is available. These functions can be used in mathematical expressions evaluated by the expression evaluator. This way, the analysis results can be grabbed and used at in the evaluation of the objective and constraint functions and their derivatives.

An example is the expression evaluator's function **noddisp** that returns a specific component of specific nodal displacement after a specific increment.

Before the appropriate expression evaluator's functions are used, the analysis result file must be connected with the pre-defined file variable *anoutfile* and the **initoutput** command must be executed.

There is a set of file interpreter's functions that read whole fields from analysis result files and store them in field variables. These functions can be used when a lot of individual field components are involved in calculation of the functions derived from analysis result. Reading whole fields into field variables and then operating with components of these variables can in such cases be quicker than using expression evaluator's functions that return individual components of the result fields. Sometimes whole result fields are used as an input for another analysis. For example, at shape optimisation the parametrised mesh, which depends on optimisation (shape) parameters, is often obtained by elastic deformation of a reference mesh. The initial mesh at specific parameters is obtained by adding displacements from the elastic analysis to the reference mesh.

12.1.1 The expression evaluator's functions that return Elfen's result data from the output file

12.1.1.1 anerror *[ ]*

Returns 1 if error at analysis occurred according to what is written in the analysis output file. Otherwise it returns 0.

12.1.1.2 ansuccess *[ ]*

Returns 1 if analysis has successfully completed according to what is written in the analysis output file. Otherwise it returns 0.

12.1.1.3 nodcoord *[node,coord]*

Returns the coordinate *coord* of the initial position of the node *node*.

12.1.1.4 nodtemp *[inc,nod,1]*

Returns the temperature of the node node *nod* after the increment *inc*. The third argument must be specified, even if it does not have any meaning. It is usually set to 1.

12.1.1.5 **noddisp** *[inc,nod,comp]*

Returns the component *comp* of the nodal displacement of the node *nod* after the increment *inc*.

12.1.1.6 **locnoddisp** *[inc,nod,comp]*

Returns the component *comp* of the nodal displacement of the node *nod* after the increment *inc* in the local coordinate system.

12.1.1.7 **nodreac** *[inc,nod,comp]*

Returns the component *comp* of the nodal reaction of the node *nod* after the increment *inc*.

12.1.1.8 **locnodreac** *[inc,nod,comp]*

Returns the component *comp* of the nodal reaction of the node *nod* after the increment *inc* in the local coordinate system.

12.1.1.9 **nodstrain** *[inc,nod,i,j]*

Returns the *i*,*j*-th component of the nodal strain in the node *nod* after the increment *inc*.

12.1.1.10 **nodstress** *[inc,nod,i,j]*

Returns the *i*,*j*-th component of the nodal stress in the node *nod* after the increment *inc*.

12.1.1.11 **nodcontforc** *[inc,nod,comp]*

Returns the component *comp* of the contact nodal force in the node *nod* after the increment *inc*.

12.1.1.12 **nodloccontforc** *[inc,nod,comp]*

Returns the component *comp* of the contact nodal force in the node *nod* after the increment *inc* in the local coordinate system.

12.1.1.13 **nodwear** *[inc,nod,comp]*

Returns specific parameters of wear in the node *nod* after the increment *inc*. The Stanislava Stupkiewicz's friction and wear model is used. *comp* specifies which data is to be returned.

12.1.2 The file interpreter's functions for exchanging data with Elfen's input and output files

12.1.2.1 ***freaddatfield*** *{ fieldspec fieldname searchpos <next>}*

Function searches the standard analysis input file (*aninfile*) for the first array with name *fieldname* from position *searchpos* on. If array components are found they are assigned to a programme's field variable specified by *fieldspec*. Function is adapted to reading arrays from *Elfen* control files of type **.dat*. *next* is an optional argument. If it is specified, the position of the first byte after the last array component is assigned to the expression evaluator's variable with the same name.

If *aninfile* is not defined, an error report is written to the standard output and to the programme's output file. If field variable *fieldspec* is not defined, a new field variable with name *fieldspec* and rank 0 is created. This action is reported to the standard output and to the programme's output file. If creating is not successful an error report is written to the standard output and to the programme's output file. If field variable *fieldspec* exists its contents is overwritten at function call. If an array with name *fieldname* is not found in the *aninfile* an error report is written to the standard output and to the programme's output file. If the number of arguments is not equal 3 or 4 an error report is written to the standard output and to the programme's output file.

12.1.2.2 ***freadneufield*** *{ fieldspec fieldname searchpos <next>}*

Like **freaddatfield**, only that the function is adapted to reading *Elfen* control files of type **.neu*.

12.1.2.3 ***freadresfield*** *{ fieldspec fieldname searchpos <next>}*

Like **freaddatfield**, only that the function searches the standard analyse output file (*anoutfile*) and is adapted to reading *Elfen* output files of type **.res*.

12.1.2.4 ***filereaddatfield*** *{ filespec fieldspec fieldname searchpos <next>}*

Function searches an arbitrary file specified in *filespec* for the first array with name *fieldname* from position *searchpos* on. If array components are found they are assigned to a programme's field variable specified by *fieldspec*. Function is adapted to reading arrays from *Elfen* control files of type **.dat*. *next* is an optional argument which specifies the position of the first byte after the last array component. It is assigned to the expression evaluator's variable with the same name.

If input file *filespec* is not defined, an error report is written to the standard output and to the programme's output file. If field variable *fieldspec* is not defined, a new field variable with name *fieldspec* and rank 0 is created. This action is reported to the standard output and to the programme's output file. If creating is not successful an error report is written to the standard output and to the programme's output file. If field variable *fieldspec* exists its contents is overwritten at function call. If *fieldname* is not found an error report is written to the standard output and to the programme's output file. If number of arguments is not equal 4 or 5 an error report is written to the standard output and to the programme's output file.

12.1.2.5 ***filereadneufield*** *{ filespec fieldspec fieldname searchpos <next>}*

Like **filereaddatfield**, only that the function is adapted to reading *Elfen* input files of type **.neu*.

12.1.2.6 ***filereadresfield*** *{ filespec fieldspec fieldname searchpos* <next>}

Like **filereaddatfield**, only that the function searches an arbitrary output file (*filespec*) and is adapted to reading *Elfen* output files of type **.res*.

12.1.2.7 **freplacefield** *{ fieldspec fieldname searchpos <start>}*

Searches for the first array with name *fieldname* in a standard analysis input file (*aninfile*) from position *searchpos* on and replaces its components by components stored in field variable *fieldspec*. *start* is an optional argument. If it is specified, the position of the beginning of an array of field components is assigned to the expression evaluator's variable with the same name.

Possible errors that can occur are:

- ◊ input file *aninfile* is not defined
- ◊ field variable *fieldspec* is not defined
- ◊ string *fieldname* is not found between the position *searchpos* and EOF
- ◊ number of arguments is not equal 3 or 4

In all of these cases an error report is written to the standard output and to the programme's output file.

12.1.2.8 **filereplacefield** *{ filespec fieldspec fieldname searchpos <start>}*

Searches for the first field with name *fieldname* in an arbitrary input file *filespec* from position *searchpos* on and replaces its components by components stored in field variable *fieldspec*. *start* is an optional argument. If it is specified, the position of the beginning of an array of field components is assigned to the expression evaluator's variable with the same name.

Possible errors that can occur are:

◊ input file *filespec* is not defined
◊ field variable *fieldspec* is not defined
◊ string *fieldname* is not found between position *searchpos* and EOF
◊ number of arguments is not equal 4 or 5

In all of these cases an error report is written to the standard output and to the programme's output file.

12.1.2.9 **filegetgroupnodesvector** *{ filespec vectorspec searchpos }*

Searches for the first element group (*IGROUP-Element_topology)* in the input file *filespec* (Elfen input file of type *.neu or new implicit) from position *searchpos* on and stores its node numbers to the vector variable *vectorspec*. Error reports are written to standard output and to the programme's output file. Possible causes for errors are: input file *filespec* is not defined or can't be found, vector variable *vectorspec* can not be located or created, *searchpos* is out of range, the number of arguments is not correct. Error reports are written to the standard output and to the programme's output file.

12.1.2.10 **filegetsurfnodesvector** *{ filespec vectorspec searchpos }*

Searches for the first slideline definition (*ISURF-Nodal_pointers_for_ segments)* in the input file *filespec* (Elfen control file of type *.neu or new implicit) from position *searchpos* on and stores its node numbers to the vector variable *vectorspec*. Error reports are written to standard output and to the programme's output file. Possible causes for errors are: input file *filespec* is not defined or can't be found, vector variable *vectorspec* can not be located or created, *searchpos* is out of range, the number of arguments is not correct. Error reports are written to the standard output and to the programme's output file.