System Preprocessors

Generated by Doxygen 1.5.9

Sun Oct 4 04:03:04 2009

Contents

1 SysPro: a System preProcessors library

1.1 Introduction

This is a library of preprocessors for numerical problems, that is, mappings of one numerical problem into another, presumably more simple one, of the same type. For example, scaling a linear system. The SysPro library operates in two modes:

- exhaustive mode: all possibly choices of a preprocessor are explored in sequence; this mode is set by command line options.
- intelligent mode: based on problem properties, a suitable choice for each preprocessor is made.

See Usage modes for more details.

Each preprocessor has the following structure, which is executed in the PreprocessedSolution() routine:

- a global setup is performed. This is a good place for computing problem features with the AnaMod library.
- a specific setup is performed; this can for instance disable certain preprocessor choices based on the computed problem features.
- a selection is made; this can be
  - a first choice, if the preprocessor is applied in exhaustive mode
  - an intelligent choice, if the exhaustive mode is off, and an intelligent choice routine exists
  - some default choice otherwise

In case of exhaustive mode, the following steps are executed inside a loop over all choices for this preprocessor, and possible all numerical option settings:

- the start function transforms the problem into a preprocessed problem
- if a next preprocessor is defined, it is applied; otherwise, the problem solving routine is applied (see Preprocessor declaration).
- the end function backtransforms the solution of the preprocessed problem into that of the original problem.

1.2 Topics

Use of the SysPro package

Preprocessor reporting and Tracing the preprocessors

The Linear package for SysPro

The interface to other packages

Command line options handling

Author:

Victor Eijkhout

Version:

1.3

Date:

unreleased

1.3 Change log

1.3 2008/08/20 : DeclarePreprocessor() has an extra argument for global deallocation at the end of a program run. Currently used in the ksp preprocessor.

2008/05/10 : end function now has two NumericalProblem parameters; this is necessary for freeing the recursive problem.

2007 : Accomodated the array length parameter in anamod/nmd

2 Approximate the coefficient matrix

A preconditioner need not be derived from the coefficient matrix. For instance, in the case of a higher order finite element matrix, incomplete factorization preconditioner are better derived from a linear element discretization of the same problem, since this matrix will be an M-matrix.

This preprocessor can perform the following approximations:

- symmetric: take the symmetric part of the coefficient matrix
- gustafsson: apply the Gustafsson modified element matrix transformation (see reference [GUS] below).

```
[GUS]
@article{Gu:modified_element,
author = {Ivar Gustafsson},
title = {An Incomplete Factorization Preconditioning Method
                based on Modification of Element Matrices},
journal = {BIT},
year = {1996},
volume = {36},
pages = {86--100}
}
```

3 Permutation / load balancing

Linear system solving is sensitive in several ways to permutations and load balancing applied to the system. This dependency typically comes through the preconditioner: incomplete factorizations are sensitive to permutations, and block Jacobi and Schwarz preconditioners are sensitive to load distributions, even without any permutation applied.

4 Flip the sign of a matrix

Most code for iterative methods and preconditioners assumes somewhere that the sign of a matrix is predominantely positive. Hence, we flip the sign of matrices that have no positive diagonal elements.

5 The iterative method

The iterative method is not really a transformation, but it is the last choice made in a preprocessor loop before the final solver is called. This means that no new matrix analysis is performed after applying this transformation.

6 The Linear package for SysPro

A linear system is a special case of a numerical problem. This file contains the routines for creating, deleting, and duplicating linear systems.

Definition of a linear system

Transformations for linear systems

The linear package for SysPro can use the NMD and AnaMod libraries, but does not require them. Any dependencies on NMD and Anamod should all be restricted to computec.

The mechanism of preprocessed solving through forward and backward transformations can be applied to all sorts of numerical problems. At present, we only supply code for linear system solving; see linearfile.

7 Transformations for linear systems

Flip the sign of a matrix

Eliminate fully determined (singleton) rows from a matrix

Permutation / load balancing

Approximate the coefficient matrix

Scale a linear system

The preconditioner

The iterative method

8 Definition of a linear system

A linear system has the following components:

- `A` : coefficient matrix
- `B` : a matrix from which to build a preconditioner. Often this will just be `A`.
- `rhs` : the right hand side
- `sol` : a known solution, if any; there is a boolean to indicate
- `init` : a starting guess for iterative methods whether a solution is known.
- `ctx` : a void pointer for storing an arbitrary data item; this can be used by the user.

See CreateLinearSystem(), DeleteLinearSystem(), LinearSystemSetParts(), LinearSystemGetParts(), LinearSystemInheritParts().

See also linearsolution.

9 linearsolution

There is an object to store the solution of a linear system.

The solution of a linear system is stored in a data structure that contains

- `out` : the computed output vector
- `statistics` : an NMD object. See LinearSolutionCreateStatistics().

See CreateLinearSolution(), LinearSolutionDelete(), LinearSolutionCopy(), LinearCopyNumericalSolution(),

10 Solution Statistics

The `LinearSolution` object carries an `NMD_metadata` object that contains performance measurements. This object is initially empty, so we need to build its content.

11 The linear solution context

We use the context pointer in a LinearSolution object to store diagnostics. This pointer is blindly copied in LinearSolutionCopy() (unlike the solution vector, which is replicated) so we have to be careful with deallocating.

12 Command line options handling

Being based on Petsc, SysPro can tailor its workings by commandline options. Options are handled by PreprocessorsOptionsHandling(). This routine needs to be called explicitly by the user, after all calls to DeclarePreprocessor(). Commandline options can be set from the program source by the Petsc call PetscOptionsSetValue().

The following commandline options are understood.

- `"-syspro_exhaustive"` : every preprocessor is cycled exhaustively, unless otherwise limited.
- `"-syspro_someprocessor exhaustive"` : the specified preprocessor is tested exhaustively.
- `"-syspro_someprocessor choice1,choice2,..."` : the specified preprocessor takes on the specified values. This induces cycling on only the specified preprocessor; if the `"-syspro_exhaustive"` option for exhaustive cycling of all preprocessors is given, the limited cycling takes precedence.
- `"-syspro_someprocessor not,choice1,choice2,..."` : limited cycling is setup, except that the the specified choices will not be used. (See TransformObjectsUseOnly() for details.)

- `"-syspro_someprocessor_somechoice_values v1,v2,v3,... "` : if a preprocessor choice has option values, this sets the values. This is also induces exhaustive cycling over this preprocessor. Note: unless the cycling is explicitly limited 'somechoice' (see the previous item), the exhaustive mode will cycle over all choices of this preprocessor.

Any preprocessor can declare its own option handler routine. The option names it handles can be anything, but should presumably not clash with the above formats. E.g., use `"-syspro_pc_iterative"` rather than `"-syspro_pc iterative"`.

13 The preconditioner

Choosing a preconditioner changes a linear system into a preconditioned system. However, this is not a transformation of any coefficient matrix, so this preprocessor is handled a bit differently from the previous ones.

14 Use of the SysPro package

14.1 Setup of the SysPro system

The setup of SysPro has to be done once per program run.

14.1.1 Global declarations

All use of SysPro has to be inside calls to SysProInitialize() and SysProFinalize(). These allocate and deallocate global data structures. You could place them right next to MPI_Initialize/Finalize or PetscInitialize/Finalize calls. The SysProFinalize() call can also be used to deallocate data that was constructed during preprocessor setup.

Another step in the global setup of SysPro is a call to SysProDeclareFunctions(), This declares functions that are of use to all preprocessors that will be declared later.

14.1.2 Preprocessor declaration

Preprocessors are declared with calls to DeclarePreprocessor(), which installs the setup functions and the forward / backward transformations.

Further specifications can be given for a specific preprocessor:

- DeclarePreprocessorIntelligentChoice() for installing a routine that will intelligently pick a preprocessor choice.
- see Computational modules interface for metadata category handling.

14.2 Setup and invocation

After the setup as described above, PreprocessorsOptionsHandling() can be called to provide the user with runtime control (see section Usage modes) over the workings of SysPro.

Preprocessed problem solving is activated by a call to PreprocessedProblemSolving(). This causes all declared preprocessors to be applied in sequence. Finally, the ultimately remaining problem is solved with the routine declared by PreprocessorsDeclareProblemSolver().

14.3 Usage modes

SysPro can be used in several modes:

- one can specify the exact preprocessor values (or several values);
- one can specify to test exhaustively the values of one preprocessor or all of them;
- SysPro can intelligently pick the appropriate preprocessor.

The intelligent preprocessor choice uses a model where each preprocessor has a measure of how applicable it is; the SysPro system then picks the most appropriate preprocessor from a given class. See Suitability functions.

See Command line options handling for details on specific and exhaustive testing.

15 The interface to other packages

SysPro by itself is not of a lot of use: it is a framework for tying together transformations and operations. There is also an interface for computing metadata.

Computational modules interface

15.1 Computational modules interface

SysPro has a facility for computing or retrieving metadata about the numerical problems it deals with. The interface comprises

- specification of preserved categories under preprocessor application with PreprocessorSetPreservedCategories().

THE FOLLOWING FACILITY IS DISABLED

Standard it comes with a dummy library `libsysprocompute.a` of routines that simply say "failed to compute/retrieve data". If the user has an actual computation package (such as AnaMod), then that can be interfaced by providing implementations of the routines SysProComputeQuantity() and SyspProRetrieveQuantity().

16 Context handling

In order to carry application-specific information and temporaries around, there are a few opaque handle contexts in syspro.

A NumericalProblem structure is defined to have a context. This is cloned at the application of a preprocessor, and deleted when its application is finished.

See SysProGetContextFunctions(), SysProProblemCloneContext(), SysProProblemDeleteContext().

In order to offer flexibility, there are several possibilities for user objects to be stored under opaque (`void*`) pointers.

16.1 contexts

Each preprocessor that has declared a `contextcreate` function, will create that context at the start of its traversal. This context is then globally registered with RegisterPreprocessorContext() under the name of this preprocessor, so that other preprocessor can have access to it with PreprocessorGetContext().

There is one special context, which is stored under the `solution` handle. This one is not created by default. See the "linear" package for an example of how to use it.

16.2 contexts

The `start_function` can create a context, which is input to the `end_function`. This context serves to preser data that is necessary for the inverse transformation that is applied in the `start_function`

17 Preprocessor reporting

For purposes of reporting, there are routines for retrieving the names of all preprocessors in sequence. This is the general idea:

```
  ierr = GetFirstPreprocessor(&name); CHKERRQ(ierr);
  while (name) {
   .....
```

```
    ierr = GetNextPreprocessor(&name); CHKERRQ(ierr);
  }
```

Similarly, StartRetrievingCurrentPreprocessors() and ContinueRetrievingCurrentPreprocessors() get the class and specific choice of the currently active preprocessors.

With StartRetrievingAllPreprocessors() and ContinueRetrievingAllPreprocessors() one can the classes, and in each class all defined choices.

The specific reporting funtions are:

- TabReportPreprocessors() : report on currently active preprocessors; suitable for database file output
- ReportEnabledPreprocessors() : report on non-disabled choices for a given preprocessor
- ReportSysProCallStackState() : report currently active preprocessors

18 Scale a linear system

This preprocessor can perform pointwise left, right, and symmetric scalings of a linear system by the diagonal of its coefficient matrix.

19 Eliminate fully determined (singleton) rows from a matrix

In a linear system, any variable whose matrix row has only a single element is independent of the rest of the system, in the sense that the other variables can be solved independently of it. The singleton preprocessor eliminates such rows, and performs the backsubstitution.

20 Tracing the preprocessors

The SysPro package does not by default print out anything, other than severe error messages (Petsc macro SETERRQ) that accompany an abort.

However, you can specify a trace function, which can further be tuned by specifying a trace context.

See SysProDeclareTraceFunction(), SysProDeclareTraceContext(), SysProTraceMessage().

21 Preprocessor definition

21.1 Class definition

A class of preprocessors (such as scaling, preconditioning) is defined using the function NewTransform().

21.2 Individual preprocessor definition

The individual preprocessors (left scaling, preprocessing by ILU) are defined in a function that is passed as the `specific_setup` argument to DeclarePreprocessor(). This function makes calls to NewTransformObject(), TransformObjectIntAnnotate() et cetera. See for instance file pc.c .

22 Suitability functions

The general mechanism form choosing between algorithms is that of 'suitability functions'. We associate with each specific preprocessor (for instance scaling/left) a function that returns either a fuzzy truth value (0–1) or 'unknown' (-1). See TransformObjectSetSuitabilityFunction(), TransformObjectGetSuitabilityFunction(). See also Preprocessor definition about specific preprocessor construction.

At the start of a preprocessor invocation, in PreprocessorSpecificSetup(), the suitability functions of all choices are evaluated, and the choices are marked as unsuitable (if the evaluate is zero), or ranked otherwise.

The only implemented suitability functions are for the linear problem; see Suitability functions for the linear problem.

23 Data Structure Index

23.1 Data Structures

Here are the data structures with brief descriptions:

24 File Index

24.1 File List

Here is a list of all files with brief descriptions:

25 Data Structure Documentation

25.1 LinearSolution_ Struct Reference

```
#include <linear_impl.h>
```

Data Fields

- int cookie
- Vec Out
- NMD_metadata statistics
- void ∗ ctx

25.1.1 Detailed Description

Definition at line 27 of file linear_impl.h.

25.1.2 Field Documentation

25.1.2.1 int LinearSolution_::cookie

Definition at line 28 of file linear_impl.h.

Referenced by CreateLinearSolution().

25.1.2.2 void∗ LinearSolution_::ctx

Definition at line 37 of file linear_impl.h.

Referenced by LinearDeleteNumericalSolutionContext(), LinearSolutionCopy(), LinearSolutionGetContext(), and LinearSolutionSetContext().

25.1.2.3 Vec LinearSolution_::Out

Definition at line 29 of file linear_impl.h.

Referenced by LinearSolutionCopy(), LinearSolutionDelete(), LinearSolutionGetVector(), and LinearSolutionSetVector().

25.1.2.4 NMD_metadata LinearSolution_::statistics

Definition at line 30 of file linear_impl.h.

Referenced by CreateLinearSolution(), LinearSolutionCopy(), LinearSolutionCopyStats(), LinearSolutionCreateStatistics(), LinearSolutionDelete(), and LinearSolutionGetStatistics().

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

- linear_impl.h

25.2 LinearSystem_ Struct Reference

```
#include <linear_impl.h>
```

Data Fields

- MPI_Comm comm
- void $*$ ctx
- int cookie
- int partsoriginal
- Mat A
- Mat B
- Vec Rhs
- Vec Sol
- Vec Init
- Vec Tmp
- PetscTruth known_solution
- NMD_metadata metadata

25.2.1 Detailed Description

Definition at line 18 of file linear_impl.h.

25.2.2 Field Documentation

25.2.2.1 Mat LinearSystem_::A

Definition at line 22 of file linear_impl.h.

Referenced by DeleteLinearSystem(), LinearSystemCopy(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetParts(), LinearSystemInheritParts(), and LinearSystemSetParts().

25.2.2.2 Mat LinearSystem_::B

Definition at line 22 of file linear_impl.h.

Referenced by DeleteLinearSystem(), LinearSystemCopy(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetParts(), LinearSystemInheritParts(), and LinearSystemSetParts().

25.2.2.3 MPI_Comm LinearSystem_::comm

Definition at line 19 of file linear_impl.h.

25.2.2.4 int LinearSystem_::cookie

Definition at line 20 of file linear_impl.h.

Referenced by CreateLinearSystem().

25.2.2.5 void* LinearSystem_::ctx

Definition at line 19 of file linear_impl.h.

Referenced by LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetContext(), and LinearSystemSetContext().

25.2.2.6 Vec LinearSystem_::Init

Definition at line 22 of file linear_impl.h.

Referenced by DeleteLinearSystem(), LinearSystemCopy(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetParts(), LinearSystemInheritParts(), and LinearSystemSetParts().

25.2.2.7 PetscTruth LinearSystem_::known_solution

Definition at line 23 of file linear_impl.h.

Referenced by LinearSystemCopy(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetKnownSolution(), and LinearSystemSetKnownSolution().

25.2.2.8 NMD_metadata LinearSystem_::metadata

Definition at line 24 of file linear_impl.h.

Referenced by LinearSystemCopy(), LinearSystemDuplicatePointers(), LinearSystemGetMetadata(), and LinearSystemSetMetadata().

25.2.2.9 int LinearSystem_::partsoriginal

Definition at line 21 of file linear_impl.h.

Referenced by CreateLinearSystem(), DeleteLinearSystem(), LinearSystemCopy(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemInheritParts(), and LinearSystemSetParts().

25.2.2.10 Vec LinearSystem_::Rhs

Definition at line 22 of file linear_impl.h.

Referenced by DeleteLinearSystem(), LinearSystemCopy(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetParts(), LinearSystemGetTmpVector(), LinearSystemInheritParts(), and LinearSystemSetParts().

25.2.2.11 Vec LinearSystem_::Sol

Definition at line 22 of file linear_impl.h.

Referenced by DeleteLinearSystem(), LinearSystemCopy(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetParts(), LinearSystemInheritParts(), and LinearSystemSetParts().

25.2.2.12 Vec LinearSystem_::Tmp

Definition at line 22 of file linear_impl.h.

Referenced by DeleteLinearSystem(), and LinearSystemGetTmpVector().

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

- linear_impl.h

25.3 NumericalProblem_ Struct Reference

```
#include <syspro_impl.h>
```

Data Fields

- MPI_Comm comm
- void $*$ ctx

25.3.1 Detailed Description

Definition at line 13 of file syspro_impl.h.

25.3.2 Field Documentation

25.3.2.1 MPI_Comm NumericalProblem_::comm

Definition at line 14 of file syspro_impl.h.

Referenced by create_solver(), NumericalProblemGetComm(), and specific_-distribution_choices().

25.3.2.2 void$*$ NumericalProblem_::ctx

Definition at line 14 of file syspro_impl.h.

Referenced by LinearSystemDuplicate(), LinearSystemDuplicatePointers(), and SysProProblemCloneContext().

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

- syspro_impl.h

25.4 PreprocessorsGlobalInfo_ Struct Reference

Data Fields

- PetscErrorCode(∗ problemmonitor)(NumericalProblem)
- PetscErrorCode(∗ classstaticsetup)(char ∗)
- PetscErrorCode(∗ classdynamicsetup)(char ∗, NumericalProblem)

  This routine is executed on the creation of a new preprocessor.

- PetscErrorCode(∗ classproblemcloner)(char ∗, char ∗, int, NumericalProblem, NumericalProblem)

  This routine is invoked at the start of each preprocessor class.

- PetscErrorCode(∗ computecategory)(char ∗, NumericalProblem)

  This routine is called everytime a new problem is created with a class/option pair.

- PetscErrorCode(∗ metadatacomputer)(char ∗, char ∗, Mat, void ∗, PetscTruth ∗)

  This routine is called in sequence with the names of the required metadata categories.

- PetscErrorCode(∗ clonecontext)(char ∗, char ∗, void ∗, void ∗∗)
- PetscErrorCode(∗ freecontext)(void ∗)
- PetscErrorCode(∗ problemsolver)(NumericalProblem, void ∗, NumericalSolution ∗)
- PetscErrorCode(∗ problemdelete)(NumericalProblem)
- PetscErrorCode(∗ errortracer)(NumericalProblem, NumericalSolution, char ∗)
- PetscErrorCode(∗ solutioncreator)(NumericalProblem, NumericalSolution ∗)
- PetscErrorCode(∗ solutioncopy)(NumericalSolution, NumericalSolution)
- PetscErrorCode(∗ solutiondelete)(NumericalSolution)
- PetscErrorCode(∗ solutioncontextdelete)(NumericalSolution)

25.4.1 Detailed Description

Definition at line 157 of file preprocess.c.

25.4.2 Field Documentation

25.4.2.1 PetscErrorCode(∗ PreprocessorsGlobalInfo_-::classdynamicsetup)(char ∗, NumericalProblem)

This routine is executed on the creation of a new preprocessor.

It can be used to install standard options in the preprocessor transform object.

Referenced by PreprocessedSolution(), and SysProDeclareFunctions().

25.4.2.2 PetscErrorCode($*$ PreprocessorsGlobalInfo_- ::classproblemcloner)(char $*$, char $*$, int, NumericalProblem, NumericalProblem)

This routine is invoked at the start of each preprocessor class.

It is not supposed to contain problem-dependent actions. It is useful for printing trace messages, and performing analysis on each incoming problem.

Referenced by SysProDeclareFunctions().

25.4.2.3 PetscErrorCode($*$ PreprocessorsGlobalInfo_::classstaticsetup)(char $*$)

Referenced by DeclarePreprocessor(), and SysProDeclareFunctions().

25.4.2.4 PetscErrorCode($*$ PreprocessorsGlobalInfo_::clonecontext)(char $*$, char $*$, void $*$, void $**$)

Referenced by SysProDeclareFunctions(), and SysProGetContextFunctions().

25.4.2.5 PetscErrorCode($*$ PreprocessorsGlobalInfo_::computecategory)(char $*$, NumericalProblem)

This routine is called everytime a new problem is created with a class/option pair.

It can be used to copy preserved metadata elements

Referenced by ChooseFirstTransform().

25.4.2.6 PetscErrorCode(∗ PreprocessorsGlobalInfo_::errortracer)(NumericalProblem, NumericalSolution, char ∗)

Referenced by PreprocessedProblemSolving(), PreprocessedSolution(), SysProDeclareErrorTracer(), SysProGetErrorTracer(), and SysProPreprocessorEndFunction().

25.4.2.7 PetscErrorCode(∗ PreprocessorsGlobalInfo_::freecontext)(void ∗)

Referenced by SysProDeclareFunctions(), and SysProGetContextFunctions().

25.4.2.8 PetscErrorCode(∗ PreprocessorsGlobalInfo_::metadatacomputer)(char ∗, char ∗, Mat, void ∗, PetscTruth ∗)

This routine is called in sequence with the names of the required metadata categories.

25.4.2.9 PetscErrorCode(∗ PreprocessorsGlobalInfo_::problemdelete)(NumericalProblem)

Referenced by SysProDeclareFunctions(), and SysProPreprocessorEndFunction().

25.4.2.10 PetscErrorCode(∗ PreprocessorsGlobalInfo_::problemmonitor)(NumericalProblem)

Referenced by PreprocessorSpecificSetup(), and SysProDeclareProblemMonitor().

25.4.2.11 PetscErrorCode(∗ PreprocessorsGlobalInfo_::problemsolver)(NumericalProblem, void ∗, NumericalSolution ∗)

Referenced by PreprocessedProblemSolving(), PreprocessedSolution(), and SysProDeclareFunctions().

25.4.2.12 PetscErrorCode(∗ PreprocessorsGlobalInfo_::solutioncontextdelete)(NumericalSolution)

Referenced by PreprocessedSolution(), and SysProDeclareFunctions().

25.4.2.13 PetscErrorCode(∗ PreprocessorsGlobalInfo_::solutioncopy)(NumericalSolution, NumericalSolution)

Referenced by SysProDeclareFunctions(), and SysProPreprocessorEndFunction().

25.4.2.14 PetscErrorCode(∗ PreprocessorsGlobalInfo_::solutioncreator)(NumericalProblem, NumericalSolution ∗)

Referenced by SysProDeclareFunctions(), and SysProPreprocessorEndFunction().

25.4.2.15 PetscErrorCode(∗ PreprocessorsGlobalInfo_::solutiondelete)(NumericalSolution)

Referenced by PreprocessedSolution(), SysProDeclareFunctions(), and SysProPreprocessorEndFunction().

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

- preprocess.c

25.5 SalsaTransform_ Struct Reference

```
#include <syspro_impl.h>
```

Data Fields

- char ∗ name
- PetscTruth userchoices
- SalsaTransformObject ∗ transformobjects
- int alloc_objects

- int n_objects
- int ∗ aprioriselection
- int n_annotate_c
- char ∗∗ annotations_c
- int n_annotate_i
- char ∗∗ annotations_i

25.5.1 Detailed Description

Definition at line 45 of file syspro_impl.h.

25.5.2 Field Documentation

25.5.2.1 int SalsaTransform_::alloc_objects

Definition at line 48 of file syspro_impl.h.

Referenced by NewTransform(), and NewTransformObject().

25.5.2.2 char∗∗ SalsaTransform_::annotations_c

Definition at line 49 of file syspro_impl.h.

Referenced by DeregisterTransform(), SysProDefineCharAnnotation(), and TransformCharAnnotationGetIndex().

25.5.2.3 char∗∗ SalsaTransform_::annotations_i

Definition at line 50 of file syspro_impl.h.

Referenced by DeregisterTransform(), SysProDefineIntAnnotation(), TransformIntAnnotationGetIndex(), and TransformObjectGetIntAnnotation().

25.5.2.4 int ∗ SalsaTransform_::aprioriselection

Definition at line 48 of file syspro_impl.h.

Referenced by DeregisterTransform(), NewTransform(), PreprocessorApplyAprioriSelection(), and PreprocessorSaveAprioriSelection().

25.5.2.5 int SalsaTransform_::n_annotate_c

Definition at line 49 of file syspro_impl.h.

Referenced by SysProDefineCharAnnotation(), and TransformCharAnnotationGetIndex().

25.5.2.6 int SalsaTransform_::n_annotate_i

Definition at line 50 of file syspro_impl.h.

Referenced by SysProDefineIntAnnotation(), TransformIntAnnotationGetIndex(), and TransformObjectGetIntAnnotation().

25.5.2.7 int SalsaTransform_::n_objects

Definition at line 48 of file syspro_impl.h.

Referenced by ContinueRetrievingAllPreprocessors(), DeregisterTransform(), NewTransform(), NewTransformObject(), PreprocessorApplyAprioriSelection(), PreprocessorSaveAprioriSelection(), PreprocessorSpecificSetup(), TransformGetNextUnmarkedItem(), TransformGetNUnmarked(), TransformGetObjects(), TransformObjectGetByName(), TransformObjectsGetNames(), TransformObjectsMarkAll(), TransformObjectsUnmarkAll(), TransformReportEnabled(), and TransformReportTeXTable().

25.5.2.8 char∗ SalsaTransform_::name

Definition at line 46 of file syspro_impl.h.

Referenced by NewTransform(), TransformGetName(), TransformItemDescribeLong(), TransformItemDescribeShort(), TransformItemOptionMark(), TransformObjectGetTransformName(), and TransformObjectsUseOnly().

25.5.2.9 SalsaTransformObject∗ SalsaTransform_::transformobjects

Definition at line 47 of file syspro_impl.h.

Referenced by DeregisterTransform(), NewTransform(), NewTransformObject(), PreprocessorApplyAprioriSelection(), PreprocessorSaveAprioriSelection(), PreprocessorSpecificSetup(), TransformGetNextUnmarkedItem(), TransformGetNUnmarked(), TransformGetObjects(), TransformObjectGetByName(), TransformObjectsGetNames(), TransformObjectsMarkAll(), TransformObjectsUnmarkAll(), TransformReportEnabled(), and TransformReportTeXTable().

25.5.2.10 PetscTruth SalsaTransform_::userchoices

Definition at line 46 of file syspro_impl.h.

Referenced by TransformGetUserChoices(), and TransformSetUserChoices().

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

- syspro_impl.h

25.6 SalsaTransformObject_ Struct Reference

```
#include <syspro_impl.h>
```

Data Fields

- char ∗ name
- char ∗ explanation
- SalsaTransform transform
- int n_options
- int alloc_options
- int ∗ options
- char ∗ option
- char ∗∗ optionexplanation
- int ∗ options_marked
- int active_option
- int alloc_annotate_c
- char ∗∗ annotate_c
- int alloc_annotate_i
- int ∗ annotate_i

- int marked
- PetscErrorCode(∗ suitabilityfunction)(NumericalProblem, void ∗, SuitabilityValue ∗)
- void ∗ suitabilityctx

25.6.1 Detailed Description

Definition at line 32 of file syspro_impl.h.

25.6.2 Field Documentation

25.6.2.1 int SalsaTransformObject_::active_option

Definition at line 36 of file syspro_impl.h.

25.6.2.2 int SalsaTransformObject_::alloc_annotate_c

Definition at line 37 of file syspro_impl.h.

Referenced by TransformObjectCharAnnotate().

25.6.2.3 int SalsaTransformObject_::alloc_annotate_i

Definition at line 38 of file syspro_impl.h.

Referenced by TransformObjectIntAnnotate().

25.6.2.4 int SalsaTransformObject_::alloc_options

Definition at line 35 of file syspro_impl.h.

Referenced by TransformObjectAddOption().

25.6.2.5 char∗∗ SalsaTransformObject_::annotate_c

Definition at line 37 of file syspro_impl.h.

Referenced by FreeTransformObject(), and TransformObjectCharAnnotate().

25.6.2.6 int ∗ SalsaTransformObject_::annotate_i

Definition at line 38 of file syspro_impl.h.

Referenced by FreeTransformObject(), TransformObjectGetIntAnnotation(), and TransformObjectIntAnnotate().

25.6.2.7 char ∗ SalsaTransformObject_::explanation

Definition at line 33 of file syspro_impl.h.

Referenced by TransformItemDescribeLong(), TransformObjectSetExplanation(), and TransformReportTeXTable().

25.6.2.8 int SalsaTransformObject_::marked

Definition at line 39 of file syspro_impl.h.

Referenced by PreprocessorApplyAprioriSelection(), PreprocessorSaveAprioriSelection(), TransformGetNextUnmarkedItem(), TransformGetNUnmarked(), TransformItemOptionMark(), TransformObjectGetMark(), TransformObjectMark(), and TransformObjectUnmark().

25.6.2.9 int SalsaTransformObject_::n_options

Definition at line 35 of file syspro_impl.h.

Referenced by FreeTransformObject(), TransformItemDescribeShort(), TransformItemGetFirstOption(), TransformItemGetNextOption(), TransformItemOptionMark(), TransformItemOptionsUseOnly(), TransformObjectAddOption(), TransformObjectAddOptionExplanation(), TransformObjectMark(), TransformObjectUnmark(), and TransformReportTeXTable().

25.6.2.10 char∗ SalsaTransformObject_::name

Definition at line 33 of file syspro_impl.h.

Referenced by FreeTransformObject(), NewTransformObject(), Preprocessor-SpecificSetup(), TransformGetNextUnmarkedItem(), TransformItemDescribeShort(), TransformObjectAddOption(), TransformObjectGetByName(), TransformObjectGet-Name(), TransformObjectsGetNames(), and TransformReportTeXTable().

25.6.2.11 char∗ SalsaTransformObject_::option

Definition at line 35 of file syspro_impl.h.

Referenced by TransformObjectDefineOption().

25.6.2.12 char ∗∗ SalsaTransformObject_::optionexplanation

Definition at line 35 of file syspro_impl.h.

Referenced by FreeTransformObject(), TransformObjectAddOption(), and Transfor-mObjectAddOptionExplanation().

25.6.2.13 int ∗ SalsaTransformObject_::options

Definition at line 35 of file syspro_impl.h.

Referenced by FreeTransformObject(), TransformItemDescribeLong(), Transfor-mItemGetFirstOption(), TransformItemGetNextOption(), TransformItemOption-Mark(), TransformObjectAddOption(), TransformObjectAddOptionExplanation(), and TransformReportTeXTable().

25.6.2.14 int∗ SalsaTransformObject_::options_marked

Definition at line 36 of file syspro_impl.h.

Referenced by FreeTransformObject(), TransformItemGetFirstOption(), Transfor-mItemGetNextOption(), TransformItemOptionMark(), TransformObjectAddOption(), TransformObjectMark(), and TransformObjectUnmark().

25.6.2.15 void∗ SalsaTransformObject_::suitabilityctx

Definition at line 42 of file syspro_impl.h.

Referenced by TransformObjectGetSuitabilityFunction(), and TransformObjectSet-SuitabilityFunction().

25.6.2.16 PetscErrorCode(∗ SalsaTransformObject_-::suitabilityfunction)(NumericalProblem, void ∗, SuitabilityValue ∗)

Referenced by TransformObjectGetSuitabilityFunction(), and TransformObjectSet-SuitabilityFunction().

25.6.2.17 SalsaTransform SalsaTransformObject_::transform

Definition at line 34 of file syspro_impl.h.

Referenced by NewTransformObject(), TransformObjectCharAnnotate(), TransformObjectGetIntAnnotation(), TransformObjectGetTransformName(), and TransformObjectIntAnnotate().

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

- syspro_impl.h

25.7 singleton_struct Struct Reference

Data Fields

- int n
- int t
- VecScatter extractor

25.7.1 Detailed Description

Definition at line 18 of file singleton.c.

25.7.2 Field Documentation

25.7.2.1 VecScatter singleton_struct::extractor

Definition at line 18 of file singleton.c.

Referenced by back_singleton(), and eliminate_singletons().

25.7.2.2 int singleton_struct::n

Definition at line 18 of file singleton.c.

Referenced by eliminate_singletons().

25.7.2.3 int singleton_struct::t

Definition at line 18 of file singleton.c.

Referenced by back_singleton(), and eliminate_singletons().

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

- singleton.c

25.8 SystemPreprocessor_ Struct Reference

```
#include <syspro_impl.h>
```

Data Fields

- char ∗ name
- SalsaTransform transform
- char ∗ preserved
- char ∗ required
- PetscTruth exhaustive
- PetscErrorCode(∗ setup)(NumericalProblem, SalsaTransform)
- PetscErrorCode(∗ unset)(NumericalProblem)
- PetscErrorCode(∗ ctxcreate)(NumericalProblem, void ∗∗)
- PetscErrorCode(∗ ctxdelete)(void ∗)

- PetscErrorCode(∗ start_function)(char ∗, int, PetscTruth, NumericalProblem, NumericalProblem ∗, void ∗, void ∗∗, PetscTruth ∗)
- PetscErrorCode(∗ end_function)(char ∗, PetscTruth, void ∗, void ∗, NumericalProblem, NumericalProblem, NumericalSolution, NumericalSolution)
- PetscErrorCode(∗ optionshandling)()
- PetscErrorCode(∗ intelligence)(NumericalProblem, char ∗∗, char ∗∗)

25.8.1 Detailed Description

Definition at line 17 of file syspro_impl.h.

25.8.2 Field Documentation

25.8.2.1 PetscErrorCode(∗ SystemPreprocessor_::ctxcreate)(NumericalProblem, void ∗∗)

Referenced by DeclarePreprocessor(), and PreprocessedSolution().

25.8.2.2 PetscErrorCode(∗ SystemPreprocessor_::ctxdelete)(void ∗)

Referenced by DeclarePreprocessor(), and PreprocessedSolution().

25.8.2.3 PetscErrorCode(∗ SystemPreprocessor_::end_function)(char ∗, PetscTruth, void ∗, void ∗, NumericalProblem, NumericalProblem, NumericalSolution, NumericalSolution)

Referenced by DeclarePreprocessor(), and SysProPreprocessorEndFunction().

25.8.2.4 PetscTruth SystemPreprocessor_::exhaustive

Definition at line 21 of file syspro_impl.h.

Referenced by DeclarePreprocessor(), PreprocessedSolution(), and PreprocessorsOptionsHandling().

25.8.2.5 PetscErrorCode(∗ SystemPreprocessor_::intelligence)(NumericalProblem, char ∗∗, char ∗∗)

Referenced by ChooseFirstTransform(), and DeclarePreprocessorIntelligentChoice().

25.8.2.6 char∗ SystemPreprocessor_::name

Definition at line 18 of file syspro_impl.h.

Referenced by ChooseFirstTransform(), ContinueRetrievingAllPreprocessors(), ContinueRetrievingCurrentPreprocessors(), DeclarePreprocessor(), GetNextPreprocessor(), SuccessorPreprocessor(), SysProFinalize(), and SysproPreprocessorStartFunction().

25.8.2.7 PetscErrorCode(∗ SystemPreprocessor_::optionshandling)()

Referenced by DeclarePCPreprocessor(), and PreprocessorsOptionsHandling().

25.8.2.8 char∗ SystemPreprocessor_::preserved

Definition at line 20 of file syspro_impl.h.

Referenced by PreprocessorGetPreservedCategories(), PreprocessorSetPreservedCategories(), and SysProFinalize().

25.8.2.9 char ∗ SystemPreprocessor_::required

Definition at line 20 of file syspro_impl.h.

Referenced by ChooseFirstTransform(), and DeclarePreprocessorRequiredCategories().

25.8.2.10 PetscErrorCode(∗ SystemPreprocessor_::setup)(NumericalProblem, SalsaTransform)

Referenced by DeclarePreprocessor(), and PreprocessorSpecificSetup().

25.8.2.11 PetscErrorCode(∗ SystemPreprocessor_::start_function)(char ∗, int, PetscTruth, NumericalProblem, NumericalProblem ∗, void ∗, void ∗∗, PetscTruth ∗)

Referenced by DeclarePreprocessor(), and SysproPreprocessorStartFunction().

25.8.2.12 SalsaTransform SystemPreprocessor_::transform

Definition at line 19 of file syspro_impl.h.

Referenced by ContinueRetrievingAllPreprocessors(), DeclarePreprocessor(), PreprocessedSolution(), PreprocessorApplyAprioriSelection(), PreprocessorSaveAprioriSelection(), SysProFinalize(), and TransformGetByName().

25.8.2.13 PetscErrorCode(∗ SystemPreprocessor_::unset)(NumericalProblem)

Referenced by DeclarePreprocessor(), and PreprocessedSolution().

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

- syspro_impl.h

26 File Documentation

26.1 approximating.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "petsc.h"
#include "petscis.h"
```

```
#include "syspro.h"
#include "sysprotransform.h"
#include "sysprolinear.h"
#include "linear_impl.h"
```

Defines

- #define PREPROCESSOR "approximation"

Functions

- static PetscErrorCode MatSymmetricPart (NumericalProblem inproblem, NumericalProblem outproblem)
- static PetscErrorCode MatGustafssonMod (NumericalProblem inproblem, NumericalProblem outproblem)
- static PetscErrorCode setup_approximation_choices ()
- static PetscErrorCode specific_approximation_choices (NumericalProblem inproblem, SalsaTransform transform)
- static PetscErrorCode approximate_system (char ∗type, int nopt, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode unapproximate_system (char ∗type, PetscTruth overwrite, void ∗gctx, void ∗ctx, NumericalProblem problem, NumericalProblem nextproblem, NumericalSolution before, NumericalSolution after)
- PetscErrorCode DeclareApproximationPreprocessor (void)

26.1.1 Detailed Description

Definition in file approximating.c.

26.1.2 Define Documentation

26.1.2.1 #define PREPROCESSOR "approximation"

Definition at line 39 of file approximating.c.

Referenced by DeclareApproximationPreprocessor(), DeclareDistributionPreprocessor(), DeclareFlipsignPreprocessor(), DeclareKSPPreprocessor(), DeclarePCPreprocessor(), DeclareScalingPreprocessor(), DeclareSingletonPreprocessor(), pcoptionshandling(), setup_approximation_choices(), setup_distribution_choices(), setup_flipsign_choices(), setup_ksp_choices(), setup_pc_choices(), setup_scaling_choices(), setup_singleton_choices(), specific_approximation_choices(), specific_flipsign_-choices(), specific_singleton_choices(), and unset_ksps().

26.1.3 Function Documentation

26.1.3.1 static PetscErrorCode approximate_system (char $*$ *type*, int *nopt*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem $*$ *outproblem*, void $*$ *gctx*, void $**$ *ctx*, PetscTruth $*$ *success*) `[static]`

Definition at line 220 of file approximating.c.

References CHKERRQ(), ierr, LinearSystemDuplicatePointers(), LinearSystemGet-Parts(), MatGustafssonMod(), and MatSymmetricPart().

Referenced by DeclareApproximationPreprocessor().

26.1.3.2 PetscErrorCode DeclareApproximationPreprocessor (void)

Definition at line 280 of file approximating.c.

References approximate_system(), CHKERRQ(), DeclarePreprocessor(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_approximation_-choices(), specific_approximation_choices(), and unapproximate_system().

26.1.3.3 static PetscErrorCode MatGustafssonMod (NumericalProblem *inproblem*, NumericalProblem *outproblem*) `[static]`

Definition at line 92 of file approximating.c.

References CHKERRQ(), ierr, LinearSystemGetParts(), and LinearSystemSetParts().

Referenced by approximate_system().

26.1.3.4 static PetscErrorCode MatSymmetricPart (NumericalProblem *inproblem*, NumericalProblem *outproblem*) `[static]`

Definition at line 44 of file approximating.c.

References CHKERRQ(), ierr, LinearSystemGetParts(), LinearSystemSetParts(), SysProComputeQuantity(), and SysProRetrieveQuantity().

Referenced by approximate_system().

26.1.3.5 static PetscErrorCode setup_approximation_choices () `[static]`

Definition at line 146 of file approximating.c.

References CHKERRQ(), ierr, NewTransformObject(), PREPROCESSOR, TransformGetByName(), and TransformObjectSetExplanation().

Referenced by DeclareApproximationPreprocessor().

26.1.3.6 static PetscErrorCode specific_approximation_choices (NumericalProblem *inproblem*, SalsaTransform *transform*) `[static]`

This is the 'specific setup' phase of the approximation preprocessor. See Usage modes for details.

This routine eliminates the Gustafsson approximation for diagonally dominant systems, and the symmetric for symmetric systems.

Definition at line 178 of file approximating.c.

References CHKERRQ(), ierr, LinearSystemGetParts(), PREPROCESSOR, SysProRetrieveQuantity(), TransformObjectGetByName(), and TransformObjectMark().

Referenced by DeclareApproximationPreprocessor().

26.1.3.7 static PetscErrorCode unapproximate_system (char $*$ *type*, PetscTruth *overwrite*, void $*$ *gctx*, void $*$ *ctx*, NumericalProblem *problem*, NumericalProblem *nextproblem*, NumericalSolution *before*, NumericalSolution *after*) `[static]`

Definition at line 265 of file approximating.c.

References CHKERRQ(), DeleteLinearSystem(), ierr, and LinearSolutionCopy().

Referenced by DeclareApproximationPreprocessor().

26.2 compute.c File Reference

System/Anamod and NMD interface.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "petscmat.h"
#include "syspro.h"
#include "syspro_impl.h"
```

Functions

- PetscErrorCode PreprocessorSetPreservedCategories (char ∗preprocess, char ∗cats)
- PetscErrorCode PreprocessorGetPreservedCategories (char ∗preprocess, char ∗∗cats)
- PetscErrorCode DeclarePreprocessorRequiredCategories (char ∗name, char ∗required)

26.2.1 Detailed Description

System/Anamod and NMD interface.

Definition in file compute.c.

26.2.2 Function Documentation

26.2.2.1 PetscErrorCode DeclarePreprocessorRequiredCategories (char ∗ *name*, char ∗ *required*)

Indicate which metadata categories need to be computed for a successful application of this preprocessor.

Arguments:

- `name` : name of the current preprocessor
- `required` : comma-separated list of metadata categories

Definition at line 73 of file compute.c.

References CHKERRQ(), ierr, SystemPreprocessor_::required, and SystemPreprocessorGetByName().

26.2.2.2 PetscErrorCode PreprocessorGetPreservedCategories (char ∗ *preprocess*, char ∗∗ *cats*)

Definition at line 53 of file compute.c.

References CHKERRQ(), ierr, SystemPreprocessor_::preserved, and SystemPreprocessorGetByName().

26.2.2.3 PetscErrorCode PreprocessorSetPreservedCategories (char ∗ *preprocess*, char ∗ *cats*)

Definition at line 32 of file compute.c.

References CHKERRQ(), ierr, SystemPreprocessor_::preserved, and SystemPreprocessorGetByName().

Referenced by DeclareApproximationPreprocessor(), DeclareDistributionPreprocessor(), DeclareFlipsignPreprocessor(), DeclareKSPPreprocessor(), DeclarePCPreprocessor(), DeclareScalingPreprocessor(), and DeclareSingletonPreprocessor().

26.3 distribution.c File Reference

```
#include <stdlib.h>
#include "petscmat.h"
#include "petscconf.h"
#include "syspro.h"
#include "sysprotransform.h"
#include "sysprolinear.h"
#include "linear_impl.h"
#include "anamod.h"
```

Defines

- #define PREPROCESSOR "distribution"

Functions

- int SpectrumComputeUnpreconditionedSpectrum ()
- static PetscErrorCode setup_distribution_choices ()
- static PetscErrorCode specific_distribution_choices (NumericalProblem problem, SalsaTransform tf)
- static PetscErrorCode sans_partition (char ∗type, NumericalProblem inproblem, int nparts, IS ∗local_to_global, VecScatter ∗perm)
- static PetscErrorCode distribute_system (char ∗type, int nopt, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode undistribute_system (char ∗scaling_type, PetscTruth overwrite, void ∗gctx, void ∗ctx, NumericalProblem problem, NumericalProblem nextproblem, NumericalSolution before, NumericalSolution after)
- PetscErrorCode DeclareDistributionPreprocessor (void)

26.3.1 Detailed Description

Definition in file distribution.c.

26.3.2 Define Documentation

26.3.2.1 #define PREPROCESSOR "distribution"

Definition at line 20 of file distribution.c.

26.3.3 Function Documentation

26.3.3.1 PetscErrorCode DeclareDistributionPreprocessor (void)

Definition at line 299 of file distribution.c.

References CHKERRQ(), DeclarePreprocessor(), distribute_system(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_distribution_choices(), specific_distribution_choices(), and undistribute_system().

26.3.3.2 static PetscErrorCode distribute_system (char ∗ *type*, int *nopt*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem ∗ *outproblem*, void ∗ *gctx*, void ∗∗ *ctx*, PetscTruth ∗ *success*) `[static]`

Definition at line 186 of file distribution.c.

References CHKERRQ(), ierr, LinearSystemDuplicatePointers(), LinearSystemGetParts(), LinearSystemSetParts(), and sans_partition().

Referenced by DeclareDistributionPreprocessor().

26.3.3.3 static PetscErrorCode sans_partition (char ∗ *type*, NumericalProblem *inproblem*, int *nparts*, IS ∗ *local_to_global*, VecScatter ∗ *perm*) `[static]`

Definition at line 91 of file distribution.c.

References CHKERRQ(), ierr, LinearSystemGetParts(), and SysProComputeQuantity().

Referenced by distribute_system().

26.3.3.4 static PetscErrorCode setup_distribution_choices () `[static]`

Definition at line 25 of file distribution.c.

References CHKERRQ(), ierr, NewTransformObject(), PREPROCESSOR, SysProDefineIntAnnotation(), TransformGetByName(), TransformObjectIntAnnotate(), and TransformObjectSetExplanation().

Referenced by DeclareDistributionPreprocessor().

26.3.3.5 static PetscErrorCode specific_distribution_choices (NumericalProblem *problem*, SalsaTransform *tf*) `[static]`

Definition at line 69 of file distribution.c.

References CHKERRQ(), NumericalProblem_::comm, ierr, TransformGetObjects(), TransformObjectGetIntAnnotation(), and TransformObjectMark().

Referenced by DeclareDistributionPreprocessor().

26.3.3.6 int SpectrumComputeUnpreconditionedSpectrum ()

26.3.3.7 static PetscErrorCode undistribute_system (char ∗ *scaling_type*, PetscTruth *overwrite*, void ∗ *gctx*, void ∗ *ctx*, NumericalProblem *problem*, NumericalProblem *nextproblem*, NumericalSolution *before*, NumericalSolution *after*) `[static]`

Definition at line 269 of file distribution.c.

References CHKERRQ(), DeleteLinearSystem(), ierr, LinearSolutionCopyStats(), and LinearSolutionGetVector().

Referenced by DeclareDistributionPreprocessor().

26.4 flipsign.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "petsc.h"
#include "syspro.h"
#include "sysprotransform.h"
#include "sysprolinear.h"
#include "linear_impl.h"
#include "anamod.h"
```

Defines

- #define PREPROCESSOR "flipsign"

Functions

- static PetscErrorCode flipsign (char ∗type, int nopt, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode back_flipsign (char ∗flipsign_type, PetscTruth overwrite, void ∗gctx, void ∗ctx, NumericalProblem nextproblem, NumericalProblem problem, NumericalSolution flipped, NumericalSolution straight)

- static PetscErrorCode setup_flipsign_choices ()
- static PetscErrorCode specific_flipsign_choices (NumericalProblem theproblem, SalsaTransform flipsign)
- PetscErrorCode DeclareFlipsignPreprocessor (void)

26.4.1 Detailed Description

Definition in file flipsign.c.

26.4.2 Define Documentation

26.4.2.1 #define PREPROCESSOR "flipsign"

Definition at line 19 of file flipsign.c.

26.4.3 Function Documentation

26.4.3.1 static PetscErrorCode back_flipsign (char ∗ *flipsign_type*, PetscTruth *overwrite*, void ∗ *gctx*, void ∗ *ctx*, NumericalProblem *nextproblem*, NumericalProblem *problem*, NumericalSolution *flipped*, NumericalSolution *straight*) `[static]`

Definition at line 85 of file flipsign.c.

References CHKERRQ(), DeleteLinearSystem(), ierr, and LinearSolutionCopy().

Referenced by DeclareFlipsignPreprocessor().

26.4.3.2 PetscErrorCode DeclareFlipsignPreprocessor (void)

Definition at line 168 of file flipsign.c.

References back_flipsign(), CHKERRQ(), DeclarePreprocessor(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_flipsign_choices(), and specific_flipsign_choices().

26.4.3.3 static PetscErrorCode flipsign (char $*$ *type*, int *nopt*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem $*$ *outproblem*, void $*$ *gctx*, void $**$ *ctx*, PetscTruth $*$ *success*) `[static]`

Definition at line 24 of file flipsign.c.

References CHKERRQ(), ierr, LinearSystemDuplicatePointers(), LinearSystemGetParts(), LinearSystemSetParts(), and SysProComputeQuantity().

26.4.3.4 static PetscErrorCode setup_flipsign_choices () `[static]`

This routine is only called when the flipsign preprocessor is created by DeclarePreprocessor() inside DeclareFlipsignPreprocessor()

Definition at line 114 of file flipsign.c.

References CHKERRQ(), ierr, NewTransformObject(), PREPROCESSOR, TransformGetByName(), and TransformObjectSetExplanation().

Referenced by DeclareFlipsignPreprocessor().

26.4.3.5 static PetscErrorCode specific_flipsign_choices (NumericalProblem *theproblem*, SalsaTransform *flipsign*) `[static]`

This is the 'specific setup' phase of the flipsign preprocessor. See Usage modes for details.

It disables either the identity or the flip routine, to leave only the one applicable to this particular system.

Definition at line 143 of file flipsign.c.

References CHKERRQ(), ierr, PREPROCESSOR, SysProComputeQuantity(), TransformObjectGetByName(), and TransformObjectMark().

Referenced by DeclareFlipsignPreprocessor().

26.5 ksp.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
```

```
#include "string.h"
#include "syspro.h"
#include "sysprotransform.h"
#include "sysprolinear.h"
#include "sysprosuit.h"
#include "anamod.h"
#include "linksp.h"
#include "petscmat.h"
#include "petscpc.h"
#include "petscksp.h"
```

Defines

- #define PREPROCESSOR "ksp"

Functions

- static PetscErrorCode is_gmres_method (KSPType kspt, PetscTruth ∗f)
- static PetscErrorCode setup_ksp_choices ()
- static PetscErrorCode unset_ksps (NumericalProblem_ ∗dummy)
- static PetscErrorCode disable_ksps (NumericalProblem theproblem, SalsaTransform ksp)
- static PetscErrorCode set_ksp_options (SalsaTransformObject tf, int kspv)
- static PetscErrorCode setup_ksp (char ∗kspt, int kspv, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode unset_ksp (char ∗kspt, PetscTruth overwrite, void ∗gctx, void ∗ctx, NumericalProblem inproblem, NumericalProblem nextproblem, NumericalSolution old, NumericalSolution nnew)
- PetscErrorCode DeclareKSPPreprocessor (void)
- PetscErrorCode SysProLinearInstallCustomKSPMonitor (KSP solver)
- PetscErrorCode SysProLinearDeclareCustomKSPMonitor (PetscErrorCode(∗monitor)(KSP, int, PetscReal, void ∗), void ∗data)

Variables

- int gmrescycleid
- PetscErrorCode(∗ custommonitor)(KSP, int, PetscReal, void ∗) = NULL
- void ∗ monitordata = NULL

26.5.1 Define Documentation

26.5.1.1 #define PREPROCESSOR "ksp"

Definition at line 22 of file ksp.c.

26.5.2 Function Documentation

26.5.2.1 PetscErrorCode DeclareKSPPreprocessor (void)

Definition at line 367 of file ksp.c.

References CHKERRQ(), DeclarePreprocessor(), disable_ksps(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_ksp(), setup_ksp_choices(), unset_ksp(), and unset_ksps().

26.5.2.2 static PetscErrorCode disable_ksps (NumericalProblem *theproblem*, SalsaTransform *ksp*) `[static]`

Definition at line 165 of file ksp.c.

References CHKERRQ(), ierr, PreprocessorGetSetting(), TransformGetObjects(), TransformObjectGetByName(), TransformObjectGetIntAnnotation(), TransformObjectGetName(), TransformObjectMark(), and TransformObjectsUnmarkAll().

Referenced by DeclareKSPPreprocessor().

26.5.2.3 static PetscErrorCode is_gmres_method (KSPType *kspt*, PetscTruth $*$ *f*) `[static]`

Definition at line 27 of file ksp.c.

References CHKERRQ(), ierr, TransformObjectGetByName(), TransformObjectGetIntAnnotation(), and TRUTH.

Referenced by setup_ksp().

26.5.2.4 static PetscErrorCode set_ksp_options (SalsaTransformObject *tf*, int *kspv*) `[static]`

Definition at line 230 of file ksp.c.

References CHKERRQ(), ierr, TransformObjectGetIntAnnotation(), and TransformObjectGetName().

Referenced by setup_ksp().

26.5.2.5 static PetscErrorCode setup_ksp (char ∗ *kspt*, int *kspv*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem ∗ *outproblem*, void ∗ *gctx*, void ∗∗ *ctx*, PetscTruth ∗ *success*) `[static]`

Definition at line 298 of file ksp.c.

References CHKERRQ(), gmrescycleid, ierr, is_gmres_method(), LinearSystemDuplicatePointers(), PreprocessorGetContext(), set_ksp_options(), SysProLinearInstallCustomKSPMonitor(), and TransformObjectGetByName().

Referenced by DeclareKSPPreprocessor().

26.5.2.6 static PetscErrorCode setup_ksp_choices () `[static]`

Definition at line 40 of file ksp.c.

References CHKERRQ(), gmrescycleid, ierr, NewTransformObject(), onlyforsymmetricproblem(), PREPROCESSOR, SysProDefineCharAnnotation(), SysProDefineIntAnnotation(), TransformGetByName(), TransformObjectAddOption(), TransformObjectDefineOption(), TransformObjectIntAnnotate(), TransformObjectSetExplanation(), and TransformObjectSetSuitabilityFunction().

Referenced by DeclareKSPPreprocessor().

26.5.2.7 PetscErrorCode SysProLinearDeclareCustomKSPMonitor (PetscErrorCode(∗)(KSP, int, PetscReal, void ∗) *monitor*, void ∗ *data*)

Definition at line 404 of file ksp.c.

References custommonitor.

26.5.2.8 PetscErrorCode SysProLinearInstallCustomKSPMonitor (KSP *solver*)

Definition at line 390 of file ksp.c.

References CHKERRQ(), custommonitor, and ierr.

Referenced by setup_ksp().

26.5.2.9 static PetscErrorCode unset_ksp (char ∗ *kspt*, PetscTruth *overwrite*, void ∗ *gctx*, void ∗ *ctx*, NumericalProblem *inproblem*, NumericalProblem *nextproblem*, NumericalSolution *old*, NumericalSolution *nnew*) `[static]`

Definition at line 346 of file ksp.c.

References CHKERRQ(), DeleteLinearSystem(), ierr, and LinearSolutionCopy().

Referenced by DeclareKSPPreprocessor().

26.5.2.10 static PetscErrorCode unset_ksps (NumericalProblem_ ∗ *dummy*) `[static]`

Definition at line 150 of file ksp.c.

References CHKERRQ(), ierr, PREPROCESSOR, TransformObjectGetByName(), and TransformObjectGetSuitabilityFunction().

Referenced by DeclareKSPPreprocessor().

26.5.3 Variable Documentation

26.5.3.1 PetscErrorCode(∗ custommonitor)(KSP, int, PetscReal, void ∗) = NULL

Referenced by SysProLinearDeclareCustomKSPMonitor(), and SysProLinearInstall-CustomKSPMonitor().

26.5.3.2 int gmrescycleid

Definition at line 23 of file ksp.c.

Referenced by setup_ksp(), and setup_ksp_choices().

26.5.3.3 void∗ monitordata = NULL

Definition at line 386 of file ksp.c.

26.6 kspmonitor.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include "syspro.h"
#include "sysprolinear.h"
```

Defines

- #define ITER_STAGNATION -21
- #define ITER_DIVERGENCE -22

Functions

- static PetscErrorCode estimate_completion_from_hist (double ∗hist, int n, double rtol, int tracing, int ∗est)
- PetscErrorCode MonitorAdjustMaxit (KSP ksp, int it, PetscReal cg_err, void ∗data)

Variables

- int gmrescycleid

26.6.1 Define Documentation

26.6.1.1 #define ITER_DIVERGENCE -22

Definition at line 8 of file kspmonitor.c.

Referenced by estimate_completion_from_hist(), and MonitorAdjustMaxit().

26.6.1.2 #define ITER_STAGNATION -21

Definition at line 7 of file kspmonitor.c.

Referenced by estimate_completion_from_hist(), and MonitorAdjustMaxit().

26.6.2 Function Documentation

26.6.2.1 static PetscErrorCode estimate_completion_from_hist (double ∗ *hist*, int *n*, double *rtol*, int *tracing*, int ∗ *est*) `[static]`

Definition at line 13 of file kspmonitor.c.

References ITER_DIVERGENCE, and ITER_STAGNATION.

Referenced by MonitorAdjustMaxit().

26.6.2.2 PetscErrorCode MonitorAdjustMaxit (KSP *ksp*, int *it*, PetscReal *cg_err*, void ∗ *data*)

This routine analyzes the convergence history, and if the iterative method is still making progress, extends the maximum number of iterations.

Definition at line 83 of file kspmonitor.c.

References CHKERRQ(), estimate_completion_from_hist(), ierr, ITER_-DIVERGENCE, and ITER_STAGNATION.

26.6.3 Variable Documentation

26.6.3.1 int gmrescycleid

Definition at line 23 of file ksp.c.

Referenced by setup_ksp(), and setup_ksp_choices().

26.7 linear.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
#include "syspro_impl.h"
#include "sysprolinear.h"
#include "linear_impl.h"
#include "petscmat.h"
#include "nmd.h"
#include "anamod.h"
```

Functions

- PetscErrorCode LinearPackageSetUp ()
- PetscErrorCode CreateLinearSystem (MPI_Comm comm, LinearSystem ∗system)
- PetscErrorCode DeleteLinearSystem (LinearSystem system)
- PetscErrorCode LinearSystemSetParts (LinearSystem system, Mat A, Mat B, Vec Rhs, Vec Sol, Vec Init)
- PetscErrorCode LinearSystemInheritParts (LinearSystem system, Mat A, Mat B, Vec Rhs, Vec Sol, Vec Init)
- PetscErrorCode LinearSystemGetParts (LinearSystem system, Mat ∗A, Mat ∗B, Vec ∗Rhs, Vec ∗Sol, Vec ∗Init)
- PetscErrorCode LinearSystemSetContext (LinearSystem system, void ∗ctx)
- PetscErrorCode LinearSystemGetContext (LinearSystem system, void ∗∗ctx)
- PetscErrorCode LinearSystemSetKnownSolution (LinearSystem sys, PetscTruth sol)
- PetscErrorCode LinearSystemGetKnownSolution (LinearSystem sys, PetscTruth ∗sol)
- PetscErrorCode LinearSystemSetMetadata (LinearSystem system, NMD_metadata nmd)
- PetscErrorCode LinearSystemGetMetadata (LinearSystem system, NMD_metadata ∗nmd)
- PetscErrorCode LinearSystemGetTmpVector (LinearSystem sys, Vec ∗tmp)
- PetscErrorCode LinearSystemDuplicatePointers (LinearSystem problem, LinearSystem ∗newproblem)
- PetscErrorCode LinearSystemDuplicate (LinearSystem problem, LinearSystem ∗newproblem)
- PetscErrorCode LinearSystemCopy (LinearSystem old, LinearSystem lnew)
- PetscErrorCode CreateLinearSolution (LinearSolution ∗sol)

- PetscErrorCode LinearCreateNumericalSolution (NumericalProblem prob, NumericalSolution ∗sol)
- PetscErrorCode LinearSolutionDelete (LinearSolution sol)
- PetscErrorCode LinearDeleteNumericalSolution (NumericalSolution sol)
- PetscErrorCode LinearSolutionCopy (LinearSolution old, LinearSolution lnew)
- PetscErrorCode LinearCopyNumericalSolution (NumericalSolution old, NumericalSolution nnew)
- PetscErrorCode CreateDefaultLinearSolution (NumericalProblem problem, NumericalSolution ∗rsol)
- PetscErrorCode LinearSolutionSetVector (LinearSolution sol, Vec out)
- PetscErrorCode LinearSolutionGetVector (LinearSolution sol, Vec ∗out)
- PetscErrorCode LinearSolutionCreateStatistics (LinearSolution sol)
- PetscErrorCode LinearSolutionGetStatistics (LinearSolution sol, NMD_metadata ∗s)
- PetscErrorCode LinearSolutionCopyStats (LinearSolution in, LinearSolution out)
- PetscErrorCode LinearSolutionSetContext (LinearSolution sol, void ∗ctx)
- PetscErrorCode LinearSolutionGetContext (LinearSolution sol, void ∗∗ctx)
- PetscErrorCode LinearDeleteNumericalSolutionContext (NumericalSolution sol)
- PetscErrorCode LinearSystemTrueDistance (LinearSystem system, LinearSolution linsol, PetscReal ∗rnrm)
- PetscErrorCode LinearSystemTrueDistancePrint (NumericalProblem problem, NumericalSolution solution, char ∗caption)
- PetscErrorCode PreprocessedLinearSystemSolution (LinearSystem sys, LinearSolution ∗sol)

26.7.1 Detailed Description

Definition in file linear.c.

26.7.2 Function Documentation

26.7.2.1 PetscErrorCode CreateDefaultLinearSolution (NumericalProblem *problem*, NumericalSolution ∗ *rsol*)

Definition at line 516 of file linear.c.

References CHKERRQ(), CreateLinearSolution(), ierr, LinearSolutionSetVector(), LinearSystemGetParts(), and SYSPROCHECKVALIDLINSYS.

26.7.2.2 PetscErrorCode CreateLinearSolution (LinearSolution $*$ *sol*)

Definition at line 399 of file linear.c.

References CHKERRQ(), LinearSolution_::cookie, ierr, LINSOLCOOKIE, and LinearSolution_::statistics.

Referenced by CreateDefaultLinearSolution(), and LinearCreateNumericalSolution().

26.7.2.3 PetscErrorCode CreateLinearSystem (MPI_Comm *comm*, LinearSystem $*$ *system*)

Allocate the structure for a linear system

Definition at line 75 of file linear.c.

References CHKERRQ(), LinearSystem_::cookie, ierr, LINSYSCOOKIE, and LinearSystem_::partsoriginal.

Referenced by LinearSystemDuplicate(), LinearSystemDuplicatePointers(), and main().

26.7.2.4 PetscErrorCode DeleteLinearSystem (LinearSystem *system*)

Definition at line 90 of file linear.c.

References LinearSystem_::A, LinearSystem_::B, CHKERRQ(), ierr, LinearSystem_-::Init, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, SYSPROCHECKVALIDLINSYS, and LinearSystem_::Tmp.

Referenced by back_flipsign(), back_singleton(), unapproximate_system(), undistribute_system(), unscale_system(), unset_ksp(), and unset_pc().

26.7.2.5 PetscErrorCode LinearCopyNumericalSolution (NumericalSolution *old*, NumericalSolution *nnew*)

This routine is essentially LinearSolutionCopy(), except that it does casts of the arguments so that it can be used as the `solutioncopy` member of SysProDeclareFunctions()

Definition at line 503 of file linear.c.

References CHKERRQ(), ierr, and LinearSolutionCopy().

26.7.2.6 PetscErrorCode LinearCreateNumericalSolution (NumericalProblem *prob*, NumericalSolution $*$ *sol*)

Shell routine around CreateLinearSolution() to save you some type casting.

If the first argument is not NULL, its matrix is extracted and used to create the vector of the solution object.

Definition at line 420 of file linear.c.

References CHKERRQ(), CreateLinearSolution(), ierr, LinearSolutionSetVector(), and LinearSystemGetParts().

Referenced by main(), and solvelinear().

26.7.2.7 PetscErrorCode LinearDeleteNumericalSolution (NumericalSolution *sol*)

This is like LinearSolutionDelete(), except that the argument has been cast so that this routine can be used as the `solutiondelete` argument of SysProDeclareFunctions().

Definition at line 465 of file linear.c.

References CHKERRQ(), ierr, and LinearSolutionDelete().

Referenced by main().

26.7.2.8 PetscErrorCode LinearDeleteNumericalSolutionContext (NumericalSolution *sol*)

Definition at line 638 of file linear.c.

References LinearSolution_::ctx, and SYSPROCHECKVALIDLINSOL.

26.7.2.9 PetscErrorCode LinearPackageSetUp ()

Definition at line 66 of file linear.c.

26.7.2.10 PetscErrorCode LinearSolutionCopy (LinearSolution *old*, LinearSolution *lnew*)

Copy one linear solution object into another. This clearly only works if their vectors are similarly layed out.

The context pointer is blindly copied. We may have to think about this a bit more.

See also LinearCopyNumericalSolution().

Definition at line 484 of file linear.c.

References CHKERRQ(), LinearSolution_::ctx, ierr, LinearSolution_::Out, LinearSolution_::statistics, and SYSPROCHECKVALIDLINSOLa.

Referenced by back_flipsign(), LinearCopyNumericalSolution(), unapproximate_-system(), unset_ksp(), and unset_pc().

26.7.2.11 PetscErrorCode LinearSolutionCopyStats (LinearSolution *in*, LinearSolution *out*)

Definition at line 599 of file linear.c.

References CHKERRQ(), ierr, LinearSolution_::statistics, and SYSPROCHECK-VALIDLINSOLa.

Referenced by back_singleton(), undistribute_system(), and unscale_system().

26.7.2.12 PetscErrorCode LinearSolutionCreateStatistics (LinearSolution *sol*)

Definition at line 561 of file linear.c.

References CHKERRQ(), ierr, LinearSolution_::statistics, and SYSPROCHECK-VALIDLINSOL.

26.7.2.13 PetscErrorCode LinearSolutionDelete (LinearSolution *sol*)

Delete a linear solution.

This does not affect the context stored in the solution. That needs a special purpose routine.

See also LinearDeleteNumericalSolution().

Definition at line 447 of file linear.c.

References CHKERRQ(), ierr, LinearSolution_::Out, LinearSolution_::statistics, and SYSPROCHECKVALIDLINSOL.

Referenced by LinearDeleteNumericalSolution().

26.7.2.14 PetscErrorCode LinearSolutionGetContext (LinearSolution *sol*, void ∗∗ *ctx*)

Definition at line 628 of file linear.c.

References LinearSolution_::ctx, and SYSPROCHECKVALIDLINSOL.

26.7.2.15 PetscErrorCode LinearSolutionGetStatistics (LinearSolution *sol*, NMD_metadata ∗ *s*)

Definition at line 589 of file linear.c.

References LinearSolution_::statistics, and SYSPROCHECKVALIDLINSOL.

26.7.2.16 PetscErrorCode LinearSolutionGetVector (LinearSolution *sol*, Vec ∗ *out*)

Definition at line 545 of file linear.c.

References LinearSolution_::Out, and SYSPROCHECKVALIDLINSOL.

Referenced by back_singleton(), LinearSystemTrueDistance(), LinearSystemTrueDistancePrint(), main(), undistribute_system(), and unscale_system().

26.7.2.17 PetscErrorCode LinearSolutionSetContext (LinearSolution *sol*, void ∗ *ctx*)

Definition at line 618 of file linear.c.

References LinearSolution_::ctx, and SYSPROCHECKVALIDLINSOL.

26.7.2.18 PetscErrorCode LinearSolutionSetVector (LinearSolution *sol*, Vec *out*)

Definition at line 535 of file linear.c.

References LinearSolution_::Out, and SYSPROCHECKVALIDLINSOL.

Referenced by back_singleton(), CreateDefaultLinearSolution(), LinearCreateNumericalSolution(), and solvelinear().

26.7.2.19 PetscErrorCode LinearSystemCopy (LinearSystem *old*, LinearSystem *lnew*)

Copy the values of the components of an old linear system into a new. The new system has to have been created with LinearSystemDuplicate() because this routine assumes that the data structures are already in place.

Definition at line 356 of file linear.c.

References LinearSystem_::A, ALLPARTSNEW, LinearSystem_::B, CHKERRQ(), ierr, LinearSystem_::Init, LinearSystem_::known_solution, LinearSystem_::metadata, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYSa.

Referenced by scale_system().

26.7.2.20 PetscErrorCode LinearSystemDuplicate (LinearSystem *problem*, LinearSystem ∗ *newproblem*)

Allocate a new linear system, and create copies in it of the data structure, but not the values, of the components of the old system.

See also LinearSystemCopy().

Definition at line 311 of file linear.c.

References LinearSystem_::A, ALLPARTSNEW, LinearSystem_::B, CHKERRQ(), CreateLinearSystem(), NumericalProblem_::ctx, LinearSystem_::ctx, ierr, LinearSystem_::Init, LinearSystem_::known_solution, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYS.

Referenced by scale_system().

26.7.2.21 PetscErrorCode LinearSystemDuplicatePointers (LinearSystem *problem*, LinearSystem ∗ *newproblem*)

Allocate a new linear system and give it the components of the old by pointer duplication.

Definition at line 282 of file linear.c.

References LinearSystem_::A, LinearSystem_::B, CHKERRQ(), CreateLinearSystem(), NumericalProblem_::ctx, LinearSystem_::ctx, ierr, LinearSystem_::Init, LinearSystem_::known_solution, LinearSystem_::metadata, LinearSystem_-::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECK-VALIDLINSYS.

Referenced by approximate_system(), distribute_system(), eliminate_singletons(), flipsign(), setup_ksp(), and setup_pc().

26.7.2.22 PetscErrorCode LinearSystemGetContext (LinearSystem *system*, void ∗∗ *ctx*)

Definition at line 215 of file linear.c.

References LinearSystem_::ctx, and SYSPROCHECKVALIDLINSYS.

Referenced by eliminate_singletons().

26.7.2.23 PetscErrorCode LinearSystemGetKnownSolution (LinearSystem *sys*, PetscTruth ∗ *sol*)

Definition at line 235 of file linear.c.

References LinearSystem_::known_solution, and SYSPROCHECKVALIDLINSYS.

Referenced by LinearSystemTrueDistancePrint().

26.7.2.24 PetscErrorCode LinearSystemGetMetadata (LinearSystem *system*, NMD_metadata ∗ *nmd*)

Definition at line 255 of file linear.c.

References LinearSystem_::metadata, and SYSPROCHECKVALIDLINSYS.

Referenced by main(), SysProComputeQuantity(), SysProFreeQuantities(), and SysProRemoveQuantity().

26.7.2.25 PetscErrorCode LinearSystemGetParts (LinearSystem *system*, Mat ∗ *A*, Mat ∗ *B*, Vec ∗ *Rhs*, Vec ∗ *Sol*, Vec ∗ *Init*)

Get the matrices and vectors of the system

Definition at line 190 of file linear.c.

References LinearSystem_::A, LinearSystem_::B, LinearSystem_::Init, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYS.

Referenced by approximate_system(), back_singleton(), CreateDefaultLinearSolution(), distribute_system(), eliminate_singletons(), flipsign(), LinearCreateNumericalSolution(), LinearSystemTrueDistance(), LinearSystemTrueDistancePrint(), MatGustafssonMod(), MatSymmetricPart(), sans_partition(), scale_system(), setup_-pc(), solvelinear(), specific_approximation_choices(), SysProComputeQuantity(), SysProRetrieveQuantity(), and unset_pc().

26.7.2.26 PetscErrorCode LinearSystemGetTmpVector (LinearSystem *sys*, Vec ∗ *tmp*)

Definition at line 265 of file linear.c.

References CHKERRQ(), ierr, LinearSystem_::Rhs, SYSPROCHECKVALIDLINSYS, and LinearSystem_::Tmp.

Referenced by LinearSystemTrueDistance(), and LinearSystemTrueDistancePrint().

26.7.2.27 PetscErrorCode LinearSystemInheritParts (LinearSystem *system*, Mat *A*, Mat *B*, Vec *Rhs*, Vec *Sol*, Vec *Init*)

Declare the matrices and vectors for a linear system. Unlike in LinearSystemSetParts(), here the parts are marked as not original, so they will not be deleted in DeleteLinearSystem().

Definition at line 162 of file linear.c.

References LinearSystem_::A, ALLPARTSNEW, LinearSystem_::B, CHKERRQ(), ierr, LinearSystem_::Init, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYS.

26.7.2.28 PetscErrorCode LinearSystemSetContext (LinearSystem *system*, void ∗ *ctx*)

Definition at line 205 of file linear.c.

References LinearSystem_::ctx, and SYSPROCHECKVALIDLINSYS.

Referenced by eliminate_singletons().

26.7.2.29 PetscErrorCode LinearSystemSetKnownSolution (LinearSystem *sys*, PetscTruth *sol*)

Definition at line 225 of file linear.c.

References LinearSystem_::known_solution, and SYSPROCHECKVALIDLINSYS.

26.7.2.30 PetscErrorCode LinearSystemSetMetadata (LinearSystem *system*, NMD_metadata *nmd*)

Definition at line 245 of file linear.c.

References LinearSystem_::metadata, and SYSPROCHECKVALIDLINSYS.

Referenced by main().

26.7.2.31 PetscErrorCode LinearSystemSetParts (LinearSystem *system*, Mat *A*, Mat *B*, Vec *Rhs*, Vec *Sol*, Vec *Init*)

Declare the matrices and vectors for a linear system.

Arguments:

- `system`
- `A` : the matrix
- `B` : operator to construct the preconditioner from; if NULL, (or identical to A), A will be used
- rhs : right hand side
- sol : storage for the computed solution

- init : (optional) nontrivial starting vector for iterative solution

Definition at line 131 of file linear.c.

References LinearSystem_::A, LinearSystem_::B, CHKERRQ(), ierr, LinearSystem_-::Init, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYS.

Referenced by distribute_system(), eliminate_singletons(), flipsign(), main(), Mat-GustafssonMod(), MatSymmetricPart(), and setup_pc().

26.7.2.32 PetscErrorCode LinearSystemTrueDistance (LinearSystem *system*, LinearSolution *linsol*, PetscReal ∗ *rnrm*)

Definition at line 652 of file linear.c.

References CHKERRQ(), ierr, LinearSolutionGetVector(), LinearSystemGetParts(), and LinearSystemGetTmpVector().

Referenced by LinearSystemTrueDistancePrint().

26.7.2.33 PetscErrorCode LinearSystemTrueDistancePrint (NumericalProblem *problem*, NumericalSolution *solution*, char ∗ *caption*)

Definition at line 672 of file linear.c.

References CHKERRQ(), ierr, LinearSolutionGetVector(), LinearSystemGetKnown-Solution(), LinearSystemGetParts(), LinearSystemGetTmpVector(), LinearSys-temTrueDistance(), SYSPROCHECKVALIDLINSOL, and SYSPROCHECK-VALIDLINSYS.

26.7.2.34 PetscErrorCode PreprocessedLinearSystemSolution (LinearSystem *sys*, LinearSolution ∗ *sol*)

Definition at line 705 of file linear.c.

References CHKERRQ(), ierr, PreprocessedProblemSolving(), RegisterPreprocessor-Context(), and SYSPROCHECKVALIDLINSYS.

26.8 linear_impl.h File Reference

```
#include "petscmat.h"
```

```
#include "petscvec.h"
#include "petscksp.h"
#include "nmd.h"
#include "syspro_impl.h"
```

Data Structures

- struct LinearSystem_
- struct LinearSolution_

Defines

- #define LINSYSCOOKIE 3297
- #define LINSOLCOOKIE 3298
- #define SYSPROCHECKVALIDLINSYS(i) {SYSPROCHECKVALID(i,LINSYSCOOKIE,"linear system");}
- #define SYSPROCHECKVALIDLINSYSa(i, a) {SYSPROCHECKVALIDa(i,LINSYSCOOKIE,"linear system",a);}
- #define SYSPROCHECKVALIDLINSOL(i) {SYSPROCHECKVALID(i,LINSOLCOOKIE,"linear solution");}
- #define SYSPROCHECKVALIDLINSOLa(i, a) {SYSPROCHECKVALIDa(i,LINSOLCOOKIE,"linear solution",a);}
- #define ALLPARTSNEW (1+2+4+8+16)

26.8.1 Define Documentation

26.8.1.1 #define ALLPARTSNEW (1+2+4+8+16)

Definition at line 17 of file linear_impl.h.

Referenced by LinearSystemCopy(), LinearSystemDuplicate(), and LinearSystemInheritParts().

26.8.1.2 #define LINSOLCOOKIE 3298

Definition at line 11 of file linear_impl.h.

Referenced by CreateLinearSolution().

26.8.1.3 #define LINSYSCOOKIE 3297

Definition at line 10 of file linear_impl.h.

Referenced by CreateLinearSystem().

26.8.1.4 #define SYSPROCHECKVALIDLINSOL(i) {SYSPROCHECK-VALID(i,LINSOLCOOKIE,"linear solution");}

Definition at line 14 of file linear_impl.h.

Referenced by LinearDeleteNumericalSolutionContext(), LinearSolutionCreateStatistics(), LinearSolutionDelete(), LinearSolutionGetContext(), LinearSolutionGetStatistics(), LinearSolutionGetVector(), LinearSolutionSetContext(), LinearSolutionSetVector(), and LinearSystemTrueDistancePrint().

26.8.1.5 #define SYSPROCHECKVALIDLINSOLa(i, a) {SYSPROCHECKVALIDa(i,LINSOLCOOKIE,"linear solution",a);}

Definition at line 15 of file linear_impl.h.

Referenced by LinearSolutionCopy(), and LinearSolutionCopyStats().

26.8.1.6 #define SYSPROCHECKVALIDLINSYS(i) {SYSPROCHECK-VALID(i,LINSYSCOOKIE,"linear system");}

Definition at line 12 of file linear_impl.h.

Referenced by CreateDefaultLinearSolution(), DeleteLinearSystem(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetContext(), LinearSystemGetKnownSolution(), LinearSystemGetMetadata(), LinearSystemGetParts(), LinearSystemGetTmpVector(), LinearSystemInheritParts(), LinearSystemSetContext(), LinearSystemSetKnownSolution(), LinearSystemSetMetadata(), LinearSystemSetParts(), LinearSystemTrueDistancePrint(), and PreprocessedLinearSystemSolution().

26.8.1.7 #define SYSPROCHECKVALIDLINSYSa(i, a) {SYSPROCHECKVALIDa(i,LINSYSCOOKIE,"linear system",a);}

Definition at line 13 of file linear_impl.h.

Referenced by LinearSystemCopy().

26.9 linksp.h File Reference

```
#include "petscksp.h"
```

Functions

- PetscErrorCode SysProLinearInstallCustomKSPMonitor (KSP)
- PetscErrorCode SysProLinearDeclareCustomKSPMonitor (PetscErrorCode(∗)(KSP, int, PetscReal, void ∗), void ∗)

26.9.1 Function Documentation

26.9.1.1 PetscErrorCode SysProLinearDeclareCustomKSPMonitor (PetscErrorCode(∗)(KSP, int, PetscReal, void ∗), void ∗)

Definition at line 404 of file ksp.c.

References custommonitor.

26.9.1.2 PetscErrorCode SysProLinearInstallCustomKSPMonitor (KSP)

Definition at line 390 of file ksp.c.

References CHKERRQ(), custommonitor, and ierr.

Referenced by setup_ksp().

26.10 linpc.h File Reference

```
#include "petscpc.h"
```

Defines

- #define PCRASM "rasm"
- #define PCSILU "silu"
- #define PCBOOMERAMG "boomeramg"
- #define PCEUCLID "euclid"
- #define PCPARASAILS "parasails"
- #define PCPILUT "pilut"
- #define PCMUMPS "mumps"
- #define PCSPOOLES "spooles"
- #define PCSUPERLU "superlu"
- #define PCUMFPACK "umfpack"
- #define PCBS95 "bs95"

Functions

- PetscErrorCode SetPetscOptionsForPC (PC pc, PCType pct0, int pcv, int pcvv)
- PetscErrorCode set_preconditioner_base_matrix (PCType, Mat, Mat ∗)
- PetscErrorCode set_pc_options (PCType pct, int pcv, int pcvv)
- PetscErrorCode pc_short_string (KSPType, int, int, char ∗∗)

26.10.1 Define Documentation

26.10.1.1 #define PCBOOMERAMG "boomeramg"

Definition at line 10 of file linpc.h.

Referenced by SetPetscOptionsForPC(), and setup_pc_choices().

26.10.1.2 #define PCBS95 "bs95"

Definition at line 20 of file linpc.h.

Referenced by pc_short_string(), set_preconditioner_base_matrix(), SetPetscOptions-ForPC(), and setup_pc_choices().

26.10.1.3 #define PCEUCLID "euclid"

Definition at line 11 of file linpc.h.

Referenced by SetPetscOptionsForPC(), and setup_pc_choices().

26.10.1.4 #define PCMUMPS "mumps"

Definition at line 15 of file linpc.h.

Referenced by set_preconditioner_base_matrix(), SetPetscOptionsForPC(), and setup_pc_choices().

26.10.1.5 #define PCPARASAILS "parasails"

Definition at line 12 of file linpc.h.

Referenced by SetPetscOptionsForPC(), and setup_pc_choices().

26.10.1.6 #define PCPILUT "pilut"

Definition at line 13 of file linpc.h.

Referenced by SetPetscOptionsForPC(), and setup_pc_choices().

26.10.1.7 #define PCRASM "rasm"

Definition at line 7 of file linpc.h.

Referenced by SetPetscOptionsForPC(), and setup_pc_choices().

26.10.1.8 #define PCSILU "silu"

Definition at line 8 of file linpc.h.

Referenced by SetPetscOptionsForPC(), and setup_pc_choices().

26.10.1.9 #define PCSPOOLES "spooles"

Definition at line 16 of file linpc.h.

Referenced by set_preconditioner_base_matrix(), SetPetscOptionsForPC(), and setup_pc_choices().

26.10.1.10 #define PCSUPERLU "superlu"

Definition at line 17 of file linpc.h.

Referenced by set_preconditioner_base_matrix(), SetPetscOptionsForPC(), and setup_pc_choices().

26.10.1.11 #define PCUMFPACK "umfpack"

Definition at line 18 of file linpc.h.

Referenced by set_preconditioner_base_matrix(), SetPetscOptionsForPC(), and setup_pc_choices().

26.10.2 Function Documentation

26.10.2.1 PetscErrorCode pc_short_string (KSPType, int, int, char **)

Definition at line 111 of file pcstuff.c.

References CHKERRQ(), ierr, and PCBS95.

26.10.2.2 PetscErrorCode set_pc_options (PCType *pct*, int *pcv*, int *pcvv*)

26.10.2.3 PetscErrorCode set_preconditioner_base_matrix (PCType, Mat, Mat *)

Definition at line 258 of file pcstuff.c.

References CHKERRQ(), PCBS95, PCMUMPS, PCSPOOLES, PCSUPERLU, and PCUMFPACK.

Referenced by setup_pc().

26.10.2.4 PetscErrorCode SetPetscOptionsForPC (PC *pc*, PCType *pct0*, int *pcv*, int *pcvv*)

Definition at line 304 of file pcstuff.c.

References CHKERRQ(), PCBOOMERAMG, PCBS95, PCEUCLID, PCMUMPS, PCPARASAILS, PCPILUT, PCRASM, PCSILU, PCSPOOLES, PCSUPERLU, PCUMFPACK, and set_blocked_sub_pc().

Referenced by setup_pc().

26.11 Make.inc File Reference

26.12 options.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "syspro.h"
#include "sysprotransform.h"
#include "syspro_impl.h"
```

Defines

- #define TYPELEN 200

Functions

- PetscErrorCode PreprocessorsOptionsHandling ()

26.12.1 Define Documentation

26.12.1.1 #define TYPELEN 200

Referenced by PreprocessorsOptionsHandling().

26.12.2 Function Documentation

26.12.2.1 PetscErrorCode PreprocessorsOptionsHandling ()

Process commandline options that control the behaviour of SysPro. For more information see Command line options handling.

Definition at line 46 of file options.c.

References CHKERRQ(), SystemPreprocessor_::exhaustive, GetFirstPreprocessor(), GetNextPreprocessor(), ierr, SystemPreprocessor_::optionshandling, PreprocessorSaveAprioriSelection(), ReportEnabledPreprocessors(), SystemPreprocessorGetByName(), TransformGetByName(), TransformGetNextUnmarkedItem(), TransformGetNUnmarked(), TransformItemOptionsUseOnly(), TransformObjectGetName(), TransformObjectsUseOnly(), TransformSetUserChoices(), TRUTH, and TYPELEN.

Referenced by main().

26.13 pc.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "syspro.h"
#include "syspro_impl.h"
#include "sysprolinear.h"
#include "sysprotransform.h"
#include "linear_impl.h"
#include "linpc.h"
#include "linksp.h"
#include "petsc.h"
#include "petscmat.h"
#include "petscpc.h"
#include "petscksp.h"
#include "anamod.h"
```

Defines

- #define PREPROCESSOR "pc"

Functions

- static PetscErrorCode setup_pc_choices ()
- static PetscErrorCode disable_pcs (NumericalProblem theproblem, SalsaTransform pc)
- static PetscErrorCode pcoptionshandling ()
- static PetscErrorCode create_solver (NumericalProblem prob, void ∗∗ctx)
- static PetscErrorCode destroy_solver (void ∗ctx)
- static PetscErrorCode setup_pc (char ∗type, int pcv, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode unset_pc (char ∗type, PetscTruth overwrite, void ∗gctx, void ∗ctx, NumericalProblem thisproblem, NumericalProblem upproblem, NumericalSolution old, NumericalSolution nnew)
- PetscErrorCode DeclarePCPreprocessor (void)

26.13.1 Define Documentation

26.13.1.1 #define PREPROCESSOR "pc"

Definition at line 23 of file pc.c.

26.13.2 Function Documentation

26.13.2.1 static PetscErrorCode create_solver (NumericalProblem *prob*, void ∗∗ *ctx*) `[static]`

Create a solver and install a monitor that dynamically increases the maximum number of iterations.

Definition at line 274 of file pc.c.

References CHKERRQ(), NumericalProblem_::comm, and ierr.

Referenced by DeclarePCPreprocessor().

26.13.2.2 PetscErrorCode DeclarePCPreprocessor (void)

Definition at line 392 of file pc.c.

References CHKERRQ(), create_solver(), DeclarePreprocessor(), destroy_solver(), disable_pcs(), ierr, SystemPreprocessor_::optionshandling, pcoptionshandling(), PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_pc(), setup_pc_-choices(), SystemPreprocessorGetByName(), and unset_pc().

26.13.2.3 static PetscErrorCode destroy_solver (void ∗ *ctx*) `[static]`

Definition at line 294 of file pc.c.

References CHKERRQ(), and ierr.

Referenced by DeclarePCPreprocessor().

26.13.2.4 static PetscErrorCode disable_pcs (NumericalProblem *theproblem*, SalsaTransform *pc*) `[static]`

Definition at line 217 of file pc.c.

References CHKERRQ(), ierr, SysProRetrieveQuantity(), TransformObjectGetBy-Name(), and TransformObjectMark().

Referenced by DeclarePCPreprocessor().

26.13.2.5 static PetscErrorCode pcoptionshandling () `[static]`

Disable certain preconditioners based on commandline options.

At the moment this is only disabling of direct solvers if the user asks for iterative only.

Definition at line 240 of file pc.c.

References CHKERRQ(), ierr, PREPROCESSOR, RetrieveAllPreprocessorValues(), TransformObjectGetByName(), TransformObjectGetIntAnnotation(), and Transfor-mObjectMark().

Referenced by DeclarePCPreprocessor().

26.13.2.6 static PetscErrorCode setup_pc (char ∗ *type*, int *pcv*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem ∗ *outproblem*, void ∗ *gctx*, void ∗∗ *ctx*, PetscTruth ∗ *success*) `[static]`

Definition at line 314 of file pc.c.

References CHKERRQ(), ierr, LinearSystemDuplicatePointers(), LinearSystemGetParts(), LinearSystemSetParts(), PreprocessorGetContext(), set_preconditioner_base_-matrix(), SetPetscOptionsForPC(), and TRUTH.

Referenced by DeclarePCPreprocessor().

26.13.2.7 static PetscErrorCode setup_pc_choices () `[static]`

Definition at line 27 of file pc.c.

References CHKERRQ(), ierr, NewTransformObject(), PCBOOMERAMG, PCBS95, PCEUCLID, PCMUMPS, PCPARASAILS, PCPILUT, PCRASM, PCSILU, PC-SPOOLES, PCSUPERLU, PCUMFPACK, PREPROCESSOR, SysProDefineIntAnnotation(), TransformGetByName(), TransformObjectAddOption(), TransformObjectAddOptionExplanation(), TransformObjectDefineOption(), TransformObjectIntAnnotate(), and TransformObjectSetExplanation().

Referenced by DeclarePCPreprocessor().

26.13.2.8 static PetscErrorCode unset_pc (char ∗ *type*, PetscTruth *overwrite*, void ∗ *gctx*, void ∗ *ctx*, NumericalProblem *thisproblem*, NumericalProblem *upproblem*, NumericalSolution *old*, NumericalSolution *nnew*) `[static]`

Definition at line 369 of file pc.c.

References CHKERRQ(), DeleteLinearSystem(), ierr, LinearSolutionCopy(), and LinearSystemGetParts().

Referenced by DeclarePCPreprocessor().

26.14 pcstuff.c File Reference

```
#include <stdlib.h>
#include "petscpc.h"
#include "petscksp.h"
#include "anamod.h"
#include "linpc.h"
```

Functions

- static PetscErrorCode set_blocked_sub_pc (int pcv)
- PetscErrorCode pc_string (KSPType pct, int pcv, int pcvv, char ∗∗s)
- PetscErrorCode pc_short_string (KSPType pct, int pcv, int pcvv, char ∗∗s)
- static PetscErrorCode ilu_stats_function (PC pc, void ∗ctx)
- PetscErrorCode get_pc_stats_function (PCType pct, int(∗∗f)(PC, void ∗))
- PetscErrorCode set_preconditioner_base_matrix (PCType pct, Mat B, Mat ∗Buse)
- PetscErrorCode SetPetscOptionsForPC (PC pc, PCType pct0, int pcv, int pcvv)

26.14.1 Function Documentation

26.14.1.1 PetscErrorCode get_pc_stats_function (PCType *pct*, int(∗∗)(PC, void ∗) *f*)

Definition at line 243 of file pcstuff.c.

References CHKERRQ(), and ilu_stats_function().

26.14.1.2 static PetscErrorCode ilu_stats_function (PC *pc*, void ∗ *ctx*) [static]

Definition at line 226 of file pcstuff.c.

Referenced by get_pc_stats_function().

26.14.1.3 PetscErrorCode pc_short_string (KSPType *pct*, int *pcv*, int *pcvv*, char ∗∗ *s*)

Definition at line 111 of file pcstuff.c.

References CHKERRQ(), ierr, and PCBS95.

26.14.1.4 PetscErrorCode pc_string (KSPType *pct*, int *pcv*, int *pcvv*, char ∗∗ *s*)

Definition at line 61 of file pcstuff.c.

References CHKERRQ(), and ierr.

26.14.1.5 static PetscErrorCode set_blocked_sub_pc (int *pcv*) `[static]`

Set the parameter value for a pc; this routine is only called for parametrised pcs

Definition at line 13 of file pcstuff.c.

References CHKERRQ(), and ierr.

Referenced by SetPetscOptionsForPC().

26.14.1.6 PetscErrorCode set_preconditioner_base_matrix (PCType *pct*, Mat *B*, Mat ∗ *Buse*)

Definition at line 258 of file pcstuff.c.

References CHKERRQ(), PCBS95, PCMUMPS, PCSPOOLES, PCSUPERLU, and PCUMFPACK.

Referenced by setup_pc().

26.14.1.7 PetscErrorCode SetPetscOptionsForPC (PC *pc*, PCType *pct0*, int *pcv*, int *pcvv*)

Definition at line 304 of file pcstuff.c.

References CHKERRQ(), PCBOOMERAMG, PCBS95, PCEUCLID, PCMUMPS, PCPARASAILS, PCPILUT, PCRASM, PCSILU, PCSPOOLES, PCSUPERLU, PCUMFPACK, and set_blocked_sub_pc().

Referenced by setup_pc().

26.15 preprocess.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "petscmat.h"
#include "syspro.h"
#include "sysprotransform.h"
#include "syspro_impl.h"
```

Data Structures

- struct PreprocessorsGlobalInfo_

Defines

- #define NPREPROCESS 25

Typedefs

- typedef struct PreprocessorsGlobalInfo_ ∗ PreprocessorsGlobalInfo

Functions

- static PetscErrorCode CreateGlobalInfo ()
- PetscErrorCode SysProInitialize ()
- PetscErrorCode SysProFinalize ()
- PetscErrorCode DeclarePreprocessor (char ∗name, PetscErrorCode(∗this_preprocessor_setup)(), PetscErrorCode(∗specific_-setup)(NumericalProblem, SalsaTransform), PetscErrorCode(∗specific_-unset)(NumericalProblem), PetscErrorCode(∗global_unset)(), PetscErrorCode(∗ctxcreate)(NumericalProblem, void ∗∗), PetscErrorCode(∗ctxdelete)(void ∗), PetscErrorCode(∗start_function)(char ∗, int, PetscTruth, NumericalProblem, NumericalProblem ∗, void ∗, void ∗∗, PetscTruth ∗), PetscErrorCode(∗end_function)(char ∗, PetscTruth, void ∗, void ∗, NumericalProblem, NumericalProblem, NumericalSolution, NumericalSolution))
- PetscErrorCode SysProDeclareProblemMonitor (PetscErrorCode(∗monitor)(NumericalProblem))
- PetscErrorCode SysProDeclareErrorTracer (PetscErrorCode(∗tracer)(NumericalProblem, NumericalSolution, char ∗))
- PetscErrorCode SysProGetErrorTracer (PetscErrorCode(∗∗tracer)(NumericalProblem, NumericalSolution, char ∗))
- PetscErrorCode DeclarePreprocessorIntelligentChoice (char ∗name, PetscErrorCode(∗picker)(NumericalProblem, char ∗∗, char ∗∗))
- PetscErrorCode NumericalProblemGetComm (NumericalProblem prob, MPI_-Comm ∗comm)
- PetscErrorCode SysProDeclareFunctions (PetscErrorCode(∗classstaticsetup)(char ∗), PetscErrorCode(∗classdynamicsetup)(char ∗, NumericalProblem), PetscErrorCode(∗classproblemcloner)(char ∗, char ∗, int, NumericalProblem, NumericalProblem), PetscErrorCode(∗problemsolver)(NumericalProblem, void ∗, NumericalSolution ∗), PetscErrorCode(∗problemdelete)(NumericalProblem),

PetscErrorCode(∗solutioncreator)(NumericalProblem, NumericalSolution ∗), PetscErrorCode(∗solutioncopy)(NumericalSolution, NumericalSolution), PetscErrorCode(∗solutiondelete)(NumericalSolution), PetscErrorCode(∗ctxcloner)(char ∗, char ∗, void ∗, void ∗∗), PetscErrorCode(∗ctxfree)(void ∗), PetscErrorCode(∗solutioncontextdelete)(NumericalSolution))

- static PetscErrorCode SysProGetContextFunctions (PetscErrorCode(∗∗ctxcloner)(char ∗, char ∗, void ∗, void ∗∗), PetscErrorCode(∗∗ctxfree)(void ∗))
- static PetscErrorCode SysProProblemCloneContext (char ∗preprocessor, char ∗type, NumericalProblem in, NumericalProblem out)
- static PetscErrorCode SysProProblemDeleteContext (NumericalProblem problem)
- static PetscErrorCode RegisterPreprocessorSetting (char ∗preprocess, char ∗type, int option)
- PetscErrorCode PreprocessorGetSetting (char ∗preprocess, char ∗∗type, int ∗option)
- PetscErrorCode RetrievePreprocessorChoice (int idx, char ∗∗type, int ∗option)
- PetscErrorCode PreprocessorGetIndex (char ∗name, int ∗prenumber)
- PetscErrorCode SystemPreprocessorGetByName (char ∗name, SystemPreprocessor ∗pp)
- PetscErrorCode TransformGetByName (char ∗name, SalsaTransform ∗tf)
- PetscErrorCode RegisterPreprocessorContext (char ∗pre, void ∗ctx)
- PetscErrorCode PreprocessorGetContext (char ∗pre, void ∗∗ctx)
- static PetscErrorCode PreprocessorSpecificSetup (char ∗preprocess, NumericalProblem problem, PetscTruth user_choices)
- static PetscErrorCode SysproPreprocessorStartFunction (SystemPreprocessor preprocessor, char ∗type, int option, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗preprocessor_context, void ∗∗transform_context, PetscTruth ∗success)
- static PetscErrorCode SysProPreprocessorEndFunction (SystemPreprocessor preprocessor, char ∗pclassname, char ∗type, PetscTruth do_not_keep_original, void ∗gctx, void ∗ctx, NumericalProblem next_problem, NumericalProblem problem, NumericalSolution next_solution, NumericalSolution ∗rsolution)
- static PetscErrorCode ChooseFirstTransform (NumericalProblem problem, char ∗preprocess, PetscTruth user_choices, PetscTruth exhaustive, SalsaTransform transform, SystemPreprocessor preprocessor, SalsaTransformObject ∗transformitem, PetscTruth ∗moretransform, char ∗∗type)
- PetscErrorCode PreprocessedSolution (char ∗pclassname, NumericalProblem problem, void ∗prevctx, NumericalSolution ∗rsolution)
- PetscErrorCode PreprocessedProblemSolving (NumericalProblem problem, NumericalSolution ∗solution)

Variables

- SystemPreprocessor ∗ preprocessors = NULL
- char ∗∗ currentpreprocessors
- char ∗∗ currentchoices = NULL
- int ∗ currentoptions = NULL
- int npreprocess = 0
- int preprocesslevel
- static void ∗∗ preprocessorcontexts
- static void ∗ solutioncontext
- static PetscErrorCode(∗∗ unsetpreprocessor)()
- static PreprocessorsGlobalInfo GlobalInfo = NULL

26.15.1 Define Documentation

26.15.1.1 #define NPREPROCESS 25

Definition at line 149 of file preprocess.c.

Referenced by DeclarePreprocessor(), and SysProInitialize().

26.15.2 Typedef Documentation

26.15.2.1 typedef struct PreprocessorsGlobalInfo_∗ PreprocessorsGlobalInfo

Definition at line 189 of file preprocess.c.

26.15.3 Function Documentation

26.15.3.1 static PetscErrorCode ChooseFirstTransform (NumericalProblem *problem*, char ∗ *preprocess*, PetscTruth *user_choices*, PetscTruth *exhaustive*, SalsaTransform *transform*, SystemPreprocessor *preprocessor*, SalsaTransformObject ∗ *transformitem*, PetscTruth ∗ *moretransform*, char ∗∗ *type*) `[static]`

Definition at line 820 of file preprocess.c.

References CHKERRQ(), PreprocessorsGlobalInfo_::computecategory, ierr, SystemPreprocessor_::intelligence, SystemPreprocessor_::name, SystemPreprocessor_::required, SysProTraceMessage(), TransformGetNextUnmarkedItem(), and TransformObjectGetName().

Referenced by PreprocessedSolution().

26.15.3.2 static PetscErrorCode CreateGlobalInfo () `[static]`

Definition at line 194 of file preprocess.c.

References CHKERRQ(), and ierr.

Referenced by SysProInitialize().

26.15.3.3 PetscErrorCode DeclarePreprocessor (char ∗ *name*, PetscErrorCode(∗)() *this_preprocessor_setup*, PetscErrorCode(∗)(NumericalProblem, SalsaTransform) *specific_setup*, PetscErrorCode(∗)(NumericalProblem) *specific_unset*, PetscErrorCode(∗)() *global_unset*, PetscErrorCode(∗)(NumericalProblem, void ∗∗) *ctxcreate*, PetscErrorCode(∗)(void ∗) *ctxdelete*, PetscErrorCode(∗)(char ∗, int, PetscTruth, NumericalProblem, NumericalProblem ∗, void ∗, void ∗∗, PetscTruth ∗) *start_function*, PetscErrorCode(∗)(char ∗, PetscTruth, void ∗, void ∗, NumericalProblem, NumericalProblem, NumericalSolution, NumericalSolution) *end_function*)

Declare a preprocessor class, by specifing its various members.

The `name` argument should not contain the colon character.

Here is an explanation of the various function arguments.

`this_preprocessor_setup()` : this routine is called only once, inside this function. This is a good place for defining all the preprocessors in this class

`specific_setup(NumericalProblem,SalsaTransform)` : this is called at the start of a preprocessing stage; one could use this for computing matrix metadata.

`global_unset(void)` : this is called in SysProFinalize().

`ctx_create(NumericalProblem,void**)` : create an object that can be used for the duration of the application of this preprocessor

`ctxdelete(void*)` : delete the context again

`start_function` : this is the function that performs the forward transform of the problem. Prototype:

```
PetscErrorCode start_function
  (char              *classmember,
   int                optionvalue,
   PetscTruth         overwrite,
   NumericalProblem   problem,
   NumericalProblem  *transformedproblem,
   void              *globalcontext,
   void             **localcontext,
   PetscTruth        *success)
```

`end_function` : this is the backtransform. Its main task is copying or backtransforming the preprocessed solution to the original solution.

```
PetscErrorCode end_function
  (char              *classmember,
   PetscTruth         overwrite,
   void              *globalcontext,
   void              *localcontext,
   NumericalProblem   pproblem,
   NumericalProblem   oproblem,
   NumericalSolution  psolution,
   NumericalSolution  osolution)
```

where `pproblem` and `psolution` are the preprocessed quantities, the end function has to unprocess them and leave the result in `oproblem`, `osolution`. Actually, `oproblem` is only for reference.

Definition at line 319 of file preprocess.c.

References CHKERRQ(), PreprocessorsGlobalInfo_::classstaticsetup, SystemPreprocessor_::ctxcreate, SystemPreprocessor_::ctxdelete, SystemPreprocessor_::end_function, SystemPreprocessor_::exhaustive, ierr, SystemPreprocessor_::name, NewTransform(), NPREPROCESS, npreprocess, SystemPreprocessor_::setup, SystemPreprocessor_::start_function, SystemPreprocessor_::transform, SystemPreprocessor_::unset, and unsetpreprocessor.

Referenced by DeclareApproximationPreprocessor(), DeclareDistributionPreprocessor(), DeclareFlipsignPreprocessor(), DeclareKSPPreprocessor(), DeclarePCPreprocessor(), DeclareScalingPreprocessor(), DeclareSingletonPreprocessor(), and main().

26.15.3.4 PetscErrorCode DeclarePreprocessorIntelligentChoice (char ∗ *name*, PetscErrorCode(∗)(NumericalProblem, char ∗∗, char ∗∗) *picker*)

Install a function to pick the optimal choice for a preprocessor

Definition at line 399 of file preprocess.c.

References CHKERRQ(), ierr, SystemPreprocessor_::intelligence, and SystemPreprocessorGetByName().

Referenced by DeclareScalingPreprocessor().

26.15.3.5 PetscErrorCode NumericalProblemGetComm (NumericalProblem *prob*, MPI_Comm ∗ *comm*)

Definition at line 411 of file preprocess.c.

References NumericalProblem_::comm.

Referenced by create_solver().

26.15.3.6 PetscErrorCode PreprocessedProblemSolving (NumericalProblem *problem*, NumericalSolution ∗ *solution*)

Invoking this routine starts the preprocessing and ultimate solution of the numerical problem.

Definition at line 1063 of file preprocess.c.

References CHKERRQ(), PreprocessorsGlobalInfo_::errortracer, GetFirstPreprocessor(), ierr, PreprocessedSolution(), preprocesslevel, and PreprocessorsGlobalInfo_::problemsolver.

Referenced by main(), and PreprocessedLinearSystemSolution().

26.15.3.7 PetscErrorCode PreprocessedSolution (char ∗ *pclassname*, NumericalProblem *problem*, void ∗ *prevctx*, NumericalSolution ∗ *rsolution*)

This routine handles the application of one preprocessor. Depending on the runtime setup (see section Usage modes), one choice is applied, or a sequence of choices is applied consecutively. The forward and backward transformation of the preprocessor are done here, and if necessary, backup copies of the system are kept around.

Definition at line 868 of file preprocess.c.

References CHKERRQ(), ChooseFirstTransform(), PreprocessorsGlobalInfo_::classdynamicsetup, SystemPreprocessor_::ctxcreate, SystemPreprocessor_::ctxdelete, PreprocessorsGlobalInfo_::errortracer, SystemPreprocessor_::exhaustive, ierr, PreprocessedSolution(), preprocesslevel, PreprocessorSpecificSetup(), PreprocessorsGlobalInfo_::problemsolver, RegisterPreprocessorContext(), RegisterPreprocessorSetting(), ReportSysProCallStackState(), PreprocessorsGlobalInfo_::solutioncontextdelete, PreprocessorsGlobalInfo_::solutiondelete, SuccessorPreprocessor(), SysProPreprocessorEndFunction(), SysproPreprocessorStartFunction(), SysProProblemCloneContext(), SysProTraceMessage(), SystemPreprocessorGet-ByName(), SystemPreprocessor_::transform, TransformGetNextUnmarkedItem(), TransformGetUserChoices(), TransformItemGetFirstOption(), TransformItemGet-NextOption(), TransformObjectGetName(), TRUTH, and SystemPreprocessor_-::unset.

Referenced by PreprocessedProblemSolving(), and PreprocessedSolution().

26.15.3.8 PetscErrorCode PreprocessorGetContext (char ∗ *pre*, void ∗∗ *ctx*)

Definition at line 668 of file preprocess.c.

References CHKERRQ(), ierr, preprocessorcontexts, PreprocessorGetIndex(), and so-lutioncontext.

Referenced by setup_ksp(), and setup_pc().

26.15.3.9 PetscErrorCode PreprocessorGetIndex (char ∗ *name*, int ∗ *prenumber*)

Definition at line 577 of file preprocess.c.

References CHKERRQ(), ierr, and npreprocess.

Referenced by PreprocessorGetContext(), RegisterPreprocessorContext(), and System-PreprocessorGetByName().

26.15.3.10 PetscErrorCode PreprocessorGetSetting (char ∗ *preprocess*, char ∗∗ *type*, int ∗ *option*)

Definition at line 545 of file preprocess.c.

References currentchoices, currentoptions, currentpreprocessors, and preprocesslevel.

Referenced by disable_ksps().

26.15.3.11 static PetscErrorCode PreprocessorSpecificSetup (char ∗ *preprocess*, NumericalProblem *problem*, PetscTruth *user_choices*) `[static]`

Setup actions that are particular to a specific class of preprocessors, such as scalings. This performs the following actions:

- any user specified setup, for instance disqualifying preprocessors on purely logistical grounds.
- subsequently, the suitability functions (Suitability functions) are evaluated for each preprocessor and the current problem. In the current implementation this is only used to disqualify preprocessors. We'll get more sophisticated later.

Definition at line 697 of file preprocess.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, SalsaTransformObject_-::name, PreprocessorApplyAprioriSelection(), PreprocessorsGlobalInfo_-::problemmonitor, ReportEnabledPreprocessors(), SystemPreprocessor_::setup, SystemPreprocessorGetByName(), TransformGetByName(), TransformObjectGetSuitabilityFunction(), TransformObjectMark(), and SalsaTransform_::transformobjects.

Referenced by PreprocessedSolution().

26.15.3.12 PetscErrorCode RegisterPreprocessorContext (char ∗ *pre*, void ∗ *ctx*)

Definition at line 646 of file preprocess.c.

References CHKERRQ(), ierr, preprocessorcontexts, PreprocessorGetIndex(), and solutioncontext.

Referenced by PreprocessedLinearSystemSolution(), and PreprocessedSolution().

26.15.3.13 static PetscErrorCode RegisterPreprocessorSetting (char ∗ *preprocess*, char ∗ *type*, int *option*) `[static]`

Definition at line 533 of file preprocess.c.

References currentchoices, currentoptions, currentpreprocessors, and preprocesslevel.

Referenced by PreprocessedSolution().

26.15.3.14 PetscErrorCode RetrievePreprocessorChoice (int *idx*, char ∗∗ *type*, int ∗ *option*)

Definition at line 565 of file preprocess.c.

References currentchoices, and currentoptions.

Referenced by ContinueRetrievingCurrentPreprocessors().

26.15.3.15 PetscErrorCode SysProDeclareErrorTracer (PetscErrorCode(∗)(NumericalProblem, NumericalSolution, char ∗) *tracer*)

Definition at line 378 of file preprocess.c.

References PreprocessorsGlobalInfo_::errortracer.

26.15.3.16 PetscErrorCode SysProDeclareFunctions (PetscErrorCode(∗)(char ∗) *classstaticsetup*, PetscErrorCode(∗)(char ∗, NumericalProblem) *classdynamicsetup*, PetscErrorCode(∗)(char ∗, char ∗, int, NumericalProblem, NumericalProblem) *classproblemcloner*, PetscErrorCode(∗)(NumericalProblem, void ∗, NumericalSolution ∗) *problemsolver*, PetscErrorCode(∗)(NumericalProblem) *problemdelete*, PetscErrorCode(∗)(NumericalProblem, NumericalSolution ∗) *solutioncreator*, PetscErrorCode(∗)(NumericalSolution, NumericalSolution) *solutioncopy*, PetscErrorCode(∗)(NumericalSolution) *solutiondelete*, PetscErrorCode(∗)(char ∗, char ∗, void ∗, void ∗∗) *ctxcloner*, PetscErrorCode(∗)(void ∗) *ctxfree*, PetscErrorCode(∗)(NumericalSolution) *solutioncontextdelete*)

Install various functions

- `classstaticsetup` : this function is called on each processor as it is being created; see DeclarePreprocessor().
- `classdynamicsetup` : this function is called as any invocation of a preprocessor starts; see PreprocessedSolution();
- `classproblemcloner` : a function to clone the context : optional see Tracing the preprocessors for more details.
- `problemsolver` : the ultimate problem solver : required
- `problemdelete` : delete a problem object
- `solutioncreator` : creates a solution object; optional, but required a preprocessor has an endfunction.
- `solutioncopy` : guess what this does; optional

- `solutiondelete` : optional, but needed if solutioncopy is used
- `contextcloner` : problems can carry a context; this clones the context if a problem is copied; otherwise the pointer is simply duplicated
- `contextfree` : used to delete cloned contexts
- `solutioncontextdelete` : hm.

Definition at line 440 of file preprocess.c.

References PreprocessorsGlobalInfo_::classdynamicsetup, PreprocessorsGlobalInfo_-::classproblemcloner, PreprocessorsGlobalInfo_::classstaticsetup, PreprocessorsGlobalInfo_::clonecontext, PreprocessorsGlobalInfo_::freecontext, PreprocessorsGlobalInfo_::problemdelete, PreprocessorsGlobalInfo_-::problemsolver, PreprocessorsGlobalInfo_::solutioncontextdelete, PreprocessorsGlobalInfo_::solutioncopy, PreprocessorsGlobalInfo_::solutioncreator, and PreprocessorsGlobalInfo_::solutiondelete.

Referenced by main().

26.15.3.17 PetscErrorCode SysProDeclareProblemMonitor (PetscErrorCode(∗)(NumericalProblem) *monitor*)

Definition at line 368 of file preprocess.c.

References PreprocessorsGlobalInfo_::problemmonitor.

26.15.3.18 PetscErrorCode SysProFinalize ()

Definition at line 236 of file preprocess.c.

References CHKERRQ(), currentchoices, currentoptions, currentpreprocessors, DeregisterTransform(), ierr, SystemPreprocessor_::name, npreprocess, preprocessor-contexts, SystemPreprocessor_::preserved, SystemPreprocessor_::transform, and un-setpreprocessor.

Referenced by main().

26.15.3.19 static PetscErrorCode SysProGetContextFunctions (PetscErrorCode(∗∗)(char ∗, char ∗, void ∗, void ∗∗) *ctxcloner*, PetscErrorCode(∗∗)(void ∗) *ctxfree*) `[static]`

Definition at line 492 of file preprocess.c.

References PreprocessorsGlobalInfo_::clonecontext, and PreprocessorsGlobalInfo_-::freecontext.

Referenced by SysProProblemCloneContext(), and SysProProblemDeleteContext().

26.15.3.20 PetscErrorCode SysProGetErrorTracer (PetscErrorCode(∗∗)(NumericalProblem, NumericalSolution, char ∗) *tracer*)

Definition at line 388 of file preprocess.c.

References PreprocessorsGlobalInfo_::errortracer.

26.15.3.21 PetscErrorCode SysProInitialize ()

Allocate SysPro globals. See also SysProFinalize().

Definition at line 211 of file preprocess.c.

References CHKERRQ(), CreateGlobalInfo(), currentchoices, currentoptions, current-preprocessors, ierr, NPREPROCESS, preprocessorcontexts, and unsetpreprocessor.

Referenced by main().

26.15.3.22 static PetscErrorCode SysProPreprocessorEndFunction (SystemPreprocessor *preprocessor*, char ∗ *pclassname*, char ∗ *type*, PetscTruth *do_not_keep_original*, void ∗ *gctx*, void ∗ *ctx*, NumericalProblem *next_problem*, NumericalProblem *problem*, NumericalSolution *next_solution*, NumericalSolution ∗ *rsolution*) `[static]`

Definition at line 774 of file preprocess.c.

References CHKERRQ(), SystemPreprocessor_::end_function, PreprocessorsGlobalInfo_::errortracer, ierr, PreprocessorsGlobalInfo_::problemdelete, PreprocessorsGlobalInfo_::solutioncopy, PreprocessorsGlobalInfo_::solutioncreator, PreprocessorsGlobalInfo_::solutiondelete, and SysProProblemDeleteContext().

Referenced by PreprocessedSolution().

26.15.3.23 static PetscErrorCode SysproPreprocessorStartFunction (SystemPreprocessor *preprocessor*, char ∗ *type*, int *option*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem ∗ *outproblem*, void ∗ *preprocessor_context*, void ∗∗ *transform_context*, PetscTruth ∗ *success*) `[static]`

Definition at line 749 of file preprocess.c.

References CHKERRQ(), ierr, SystemPreprocessor_::name, and SystemPreprocessor_::start_function.

Referenced by PreprocessedSolution().

26.15.3.24 static PetscErrorCode SysProProblemCloneContext (char ∗ *preprocessor*, char ∗ *type*, NumericalProblem *in*, NumericalProblem *out*) `[static]`

Definition at line 504 of file preprocess.c.

References CHKERRQ(), NumericalProblem_::ctx, ierr, and SysProGetContextFunctions().

Referenced by PreprocessedSolution().

26.15.3.25 static PetscErrorCode SysProProblemDeleteContext (NumericalProblem *problem*) `[static]`

Definition at line 516 of file preprocess.c.

References CHKERRQ(), ierr, and SysProGetContextFunctions().

Referenced by SysProPreprocessorEndFunction().

26.15.3.26 PetscErrorCode SystemPreprocessorGetByName (char ∗ *name*, SystemPreprocessor ∗ *pp*)

Definition at line 598 of file preprocess.c.

References CHKERRQ(), ierr, and PreprocessorGetIndex().

Referenced by DeclarePCPreprocessor(), DeclarePreprocessorIntelligentChoice(), DeclarePreprocessorRequiredCategories(), PreprocessedSolution(), PreprocessorGetPreservedCategories(), PreprocessorSetPreservedCategories(), PreprocessorsOptionsHandling(), PreprocessorSpecificSetup(), and TransformGetByName().

26.15.3.27 PetscErrorCode TransformGetByName (char ∗ *name*, SalsaTransform ∗ *tf*)

Definition at line 611 of file preprocess.c.

References CHKERRQ(), ierr, SystemPreprocessorGetByName(), and SystemPreprocessor_::transform.

Referenced by NewTransformObject(), PreprocessorsOptionsHandling(), PreprocessorSpecificSetup(), ReportEnabledPreprocessors(), setup_approximation_choices(), setup_distribution_choices(), setup_flipsign_choices(), setup_ksp_choices(), setup_pc_choices(), setup_scaling_choices(), setup_singleton_choices(), SysProDefineCharAnnotation(), SysProDefineIntAnnotation(), and TransformObjectGetByName().

26.15.4 Variable Documentation

26.15.4.1 char ∗∗ currentchoices = NULL

Definition at line 151 of file preprocess.c.

Referenced by PreprocessorGetSetting(), RegisterPreprocessorSetting(), RetrievePreprocessorChoice(), SysProFinalize(), and SysProInitialize().

26.15.4.2 int∗ currentoptions = NULL

Definition at line 152 of file preprocess.c.

Referenced by PreprocessorGetSetting(), RegisterPreprocessorSetting(), RetrievePreprocessorChoice(), SysProFinalize(), and SysProInitialize().

26.15.4.3 char∗∗ currentpreprocessors

Definition at line 151 of file preprocess.c.

Referenced by PreprocessorGetSetting(), RegisterPreprocessorSetting(), SysProFinalize(), and SysProInitialize().

26.15.4.4 PreprocessorsGlobalInfo GlobalInfo = NULL `[static]`

Definition at line 190 of file preprocess.c.

26.15.4.5 int npreprocess = 0

Definition at line 153 of file preprocess.c.

Referenced by ContinueRetrievingAllPreprocessors(), DeclarePreprocessor(), GetNextPreprocessor(), PreprocessorGetIndex(), RetrieveAllPreprocessorValues(), SuccessorPreprocessor(), and SysProFinalize().

26.15.4.6 int preprocesslevel

Definition at line 153 of file preprocess.c.

Referenced by ContinueRetrievingCurrentPreprocessors(), PreprocessedProblemSolving(), PreprocessedSolution(), PreprocessorGetSetting(), and RegisterPreprocessorSetting().

26.15.4.7 void** preprocessorcontexts `[static]`

Definition at line 154 of file preprocess.c.

Referenced by PreprocessorGetContext(), RegisterPreprocessorContext(), SysProFinalize(), and SysProInitialize().

26.15.4.8 SystemPreprocessor* preprocessors = NULL

Definition at line 150 of file preprocess.c.

26.15.4.9 void * solutioncontext `[static]`

Definition at line 154 of file preprocess.c.

Referenced by PreprocessorGetContext(), and RegisterPreprocessorContext().

26.15.4.10 PetscErrorCode(∗∗ unsetpreprocessor)() `[static]`

Referenced by DeclarePreprocessor(), SysProFinalize(), and SysProInitialize().

26.16 reporting.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "syspro.h"
#include "sysprotransform.h"
#include "syspro_impl.h"
```

Defines

- #define LINELEN 1500
- #define REPOSITION(a, b)
- #define MAXLEN 500

Functions

- PetscErrorCode GetFirstPreprocessor (char ∗∗preprocess)
- PetscErrorCode GetNextPreprocessor (char ∗∗next_one)
- PetscErrorCode SuccessorPreprocessor (char ∗this_one, char ∗∗next_one)
- PetscErrorCode StartRetrievingCurrentPreprocessors (char ∗∗cclass, char ∗∗type, int ∗opt, PetscTruth ∗success)
- PetscErrorCode ContinueRetrievingCurrentPreprocessors (char ∗∗cclass, char ∗∗type, int ∗opt, PetscTruth ∗success)
- PetscErrorCode StartRetrievingAllPreprocessors (char ∗∗cclass, char ∗∗∗types, int ∗ntypes, PetscTruth ∗success)
- PetscErrorCode InitRetrievingPreprocessors ()
- PetscErrorCode ContinueRetrievingAllPreprocessors (char ∗∗cclass, char ∗∗∗types, int ∗ntypes, PetscTruth ∗success)
- PetscErrorCode RetrieveAllPreprocessorValues (char ∗cclass, char ∗∗∗types, int ∗ntypes)

- static PetscErrorCode TabReportPreprocessors (PetscTruth active, char ∗∗key, char ∗∗val, int separator)
- PetscErrorCode TabReportAllPreprocessors (char ∗∗key, int separator)
- PetscErrorCode TabReportActivePreprocessors (char ∗∗key, char ∗∗val, int separator)
- PetscErrorCode ScreenOutputTab (char ∗key, char ∗val)
- PetscErrorCode ScreenOutputTabLine (char ∗key, char ∗val)
- PetscErrorCode ReportEnabledPreprocessors (char ∗name)
- PetscErrorCode ReportSysProCallStackState (char ∗name)

Variables

- int npreprocess
- int preprocesslevel
- static int preprocessreadout
- SystemPreprocessor ∗ preprocessors

26.16.1 Define Documentation

26.16.1.1 #define LINELEN 1500

Definition at line 226 of file reporting.c.

Referenced by TabReportPreprocessors().

26.16.1.2 #define MAXLEN 500

Referenced by ScreenOutputTabLine().

26.16.1.3 #define REPOSITION(a, b)

Value:

```
ierr = PetscStrlen(a,&b); CHKERRQ(ierr); \
  if (b>LINELEN) SETERRQ(1,"string overflow")
```

Definition at line 227 of file reporting.c.

Referenced by TabReportPreprocessors().

26.16.2 Function Documentation

26.16.2.1 PetscErrorCode ContinueRetrievingAllPreprocessors (char ∗∗ *cclass*, char ∗∗∗ *types*, int ∗ *ntypes*, PetscTruth ∗ *success*)

This routine is to be used repeatedly after an initial call to StartRetrievingAllPreprocessors().

The `types` argument is allocated internally and should be deallocated by the user.

Definition at line 186 of file reporting.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, SystemPreprocessor_::name, npreprocess, preprocessreadout, SystemPreprocessor_::transform, and TransformObjectsGetNames().

Referenced by RetrieveAllPreprocessorValues(), StartRetrievingAllPreprocessors(), and TabReportPreprocessors().

26.16.2.2 PetscErrorCode ContinueRetrievingCurrentPreprocessors (char ∗∗ *cclass*, char ∗∗ *type*, int ∗ *opt*, PetscTruth ∗ *success*)

This routine is to be used repeatedly after an initial call to StartRetrievingCurrentPreprocessors().

Definition at line 125 of file reporting.c.

References CHKERRQ(), ierr, SystemPreprocessor_::name, preprocesslevel, preprocessreadout, and RetrievePreprocessorChoice().

Referenced by StartRetrievingCurrentPreprocessors(), and TabReportPreprocessors().

26.16.2.3 PetscErrorCode GetFirstPreprocessor (char ∗∗ *preprocess*)

Get the name of the first declared preprocessor (in order of declaration) or null if none have been declared. Subsequent preprocessors can be retrieved with GetNextPreprocessor() or SuccessorPreprocessor().

Definition at line 45 of file reporting.c.

References CHKERRQ(), GetNextPreprocessor(), ierr, and preprocessreadout.

Referenced by PreprocessedProblemSolving(), and PreprocessorsOptionsHandling().

26.16.2.4 PetscErrorCode GetNextPreprocessor (char ** *next_one*)

Get the next preprocessor according to the variable preprocessreadout.

The result is null if there are no further preprocessors.

Definition at line 61 of file reporting.c.

References SystemPreprocessor_::name, npreprocess, and preprocessreadout.

Referenced by GetFirstPreprocessor(), and PreprocessorsOptionsHandling().

26.16.2.5 PetscErrorCode InitRetrievingPreprocessors ()

Definition at line 169 of file reporting.c.

References preprocessreadout.

Referenced by TabReportPreprocessors().

26.16.2.6 PetscErrorCode ReportEnabledPreprocessors (char * *name*)

Report preprocessor choices that are available after the specific setup has possible disabled some of the registered ones. This function uses the `sysprotrace` function, so this has to have been declared.

Definition at line 387 of file reporting.c.

References CHKERRQ(), ierr, SysProHasTrace(), SysProTraceMessage(), TransformGetByName(), and TransformReportEnabled().

Referenced by PreprocessorsOptionsHandling(), and PreprocessorSpecificSetup().

26.16.2.7 PetscErrorCode ReportSysProCallStackState (char * *name*)

Report preprocessor choices that are available after the specific setup has possible disabled some of the registered ones. This function uses the `sysprotrace` function, so this has to have been declared.

Definition at line 411 of file reporting.c.

References CHKERRQ(), ierr, ScreenOutputTabLine(), SysProHasTrace(), SysProTraceMessage(), and TabReportPreprocessors().

Referenced by PreprocessedSolution().

26.16.2.8 PetscErrorCode RetrieveAllPreprocessorValues (char $*$ *cclass*, char $***$ *types*, int $*$ *ntypes*)

Definition at line 210 of file reporting.c.

References CHKERRQ(), ContinueRetrievingAllPreprocessors(), ierr, npreprocess, and preprocessreadout.

Referenced by pcoptionshandling().

26.16.2.9 PetscErrorCode ScreenOutputTab (char $*$ *key*, char $*$ *val*)

Definition at line 320 of file reporting.c.

References CHKERRQ(), ierr, SysProHasTrace(), and SysProTraceMessage().

26.16.2.10 PetscErrorCode ScreenOutputTabLine (char $*$ *key*, char $*$ *val*)

Definition at line 347 of file reporting.c.

References CHKERRQ(), ierr, MAXLEN, SysProHasTrace(), and SysProTraceMessage().

Referenced by ReportSysProCallStackState().

26.16.2.11 PetscErrorCode StartRetrievingAllPreprocessors (char $**$ *cclass*, char $***$ *types*, int $*$ *ntypes*, PetscTruth $*$ *success*)

This routine gives the class of the first declared preprocessor, and all possible values. To get the next preprocessor, call ContinueRetrievingAllPreprocessors().

The class, types, and ntypes arguments can all be null.

The `types` argument is allocated internally and should be deallocated by the user.

Definition at line 157 of file reporting.c.

References CHKERRQ(), ContinueRetrievingAllPreprocessors(), ierr, and preprocessreadout.

26.16.2.12 PetscErrorCode StartRetrievingCurrentPreprocessors (char $**$ *cclass*, char $**$ *type*, int $*$ *opt*, PetscTruth $*$ *success*)

This routine gives the class and current value of the first declared preprocessor. To get the next preprocessor, call ContinueRetrievingAllPreprocessors().

The class, types, and ntypes arguments can all be null.

Definition at line 108 of file reporting.c.

References CHKERRQ(), ContinueRetrievingCurrentPreprocessors(), ierr, and preprocessreadout.

26.16.2.13 PetscErrorCode SuccessorPreprocessor (char $*$ *this_one*, char $**$ *next_one*)

Given a preprocessor, get the name of the next one (in order of declaration) or null if there are no further ones.

The arguments are allowed to be the same.

Definition at line 78 of file reporting.c.

References CHKERRQ(), ierr, SystemPreprocessor_::name, and npreprocess.

Referenced by PreprocessedSolution().

26.16.2.14 PetscErrorCode TabReportActivePreprocessors (char $**$ *key*, char $**$ *val*, int *separator*)

Definition at line 309 of file reporting.c.

References CHKERRQ(), ierr, and TabReportPreprocessors().

26.16.2.15 PetscErrorCode TabReportAllPreprocessors (char $**$ *key*, int *separator*)

Definition at line 295 of file reporting.c.

References CHKERRQ(), ierr, and TabReportPreprocessors().

26.16.2.16 static PetscErrorCode TabReportPreprocessors (PetscTruth *active*, char ∗∗ *key*, char ∗∗ *val*, int *separator*) [static]

Report all defined preprocessors. Either key and val argument can be NULL. The string arguments returned need to be deallocated in the calling environment.

Definition at line 237 of file reporting.c.

References CHKERRQ(), ContinueRetrievingAllPreprocessors(), ContinueRetrievingCurrentPreprocessors(), ierr, InitRetrievingPreprocessors(), LINELEN, and REPOSITION.

Referenced by ReportSysProCallStackState(), TabReportActivePreprocessors(), and TabReportAllPreprocessors().

26.16.3 Variable Documentation

26.16.3.1 int npreprocess

Definition at line 153 of file preprocess.c.

Referenced by ContinueRetrievingAllPreprocessors(), DeclarePreprocessor(), GetNextPreprocessor(), PreprocessorGetIndex(), RetrieveAllPreprocessorValues(), SuccessorPreprocessor(), and SysProFinalize().

26.16.3.2 int preprocesslevel

Definition at line 153 of file preprocess.c.

Referenced by ContinueRetrievingCurrentPreprocessors(), PreprocessedProblemSolving(), PreprocessedSolution(), PreprocessorGetSetting(), and RegisterPreprocessorSetting().

26.16.3.3 SystemPreprocessor∗ preprocessors

Definition at line 150 of file preprocess.c.

26.16.3.4 int preprocessreadout [static]

Definition at line 34 of file reporting.c.

Referenced by ContinueRetrievingAllPreprocessors(), ContinueRetrievingCurrentPreprocessors(), GetFirstPreprocessor(), GetNextPreprocessor(), InitRetrievingPreprocessors(), RetrieveAllPreprocessorValues(), StartRetrievingAllPreprocessors(), and StartRetrievingCurrentPreprocessors().

26.17 scaling.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "petsc.h"
#include "syspro.h"
#include "sysprotransform.h"
#include "sysprolinear.h"
#include "petscmat.h"
```

Defines

- #define PREPROCESSOR "scaling"

Functions

- static PetscErrorCode set_intelligent_scaling (NumericalProblem theproblem, char ∗∗type, char ∗∗reason)
- static PetscErrorCode scale_system (char ∗type, int nopt, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode unscale_system (char ∗scaling_type, PetscTruth overwrite, void ∗gctx, void ∗ctx, NumericalProblem problem, NumericalProblem nextproblem, NumericalSolution scaled, NumericalSolution unscaled)
- static PetscErrorCode setup_scaling_choices ()
- static PetscErrorCode specific_scaling_choices (NumericalProblem theproblem, SalsaTransform scaling)
- PetscErrorCode DeclareScalingPreprocessor (void)

26.17.1 Detailed Description

Definition in file scaling.c.

26.17.2 Define Documentation

26.17.2.1 #define PREPROCESSOR "scaling"

Definition at line 16 of file scaling.c.

26.17.3 Function Documentation

26.17.3.1 PetscErrorCode DeclareScalingPreprocessor (void)

Definition at line 292 of file scaling.c.

References CHKERRQ(), DeclarePreprocessor(), DeclarePreprocessorIntelligentChoice(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), scale_system(), set_intelligent_scaling(), setup_scaling_choices(), specific_scaling_-choices(), and unscale_system().

Referenced by main().

26.17.3.2 static PetscErrorCode scale_system (char $*$ *type*, int *nopt*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem $*$ *outproblem*, void $*$ *gctx*, void $**$ *ctx*, PetscTruth $*$ *success*) `[static]`

Definition at line 67 of file scaling.c.

References CHKERRQ(), ierr, LinearSystemCopy(), LinearSystemDuplicate(), and LinearSystemGetParts().

Referenced by DeclareScalingPreprocessor().

26.17.3.3 static PetscErrorCode set_intelligent_scaling (NumericalProblem *theproblem*, char $**$ *type*, char $**$ *reason*) `[static]`

Definition at line 21 of file scaling.c.

References CHKERRQ(), ierr, SysProRetrieveQuantity(), and TRUTH.

Referenced by DeclareScalingPreprocessor().

26.17.3.4 static PetscErrorCode setup_scaling_choices () `[static]`

This routine is called by DeclarePreprocessor()

Definition at line 230 of file scaling.c.

References CHKERRQ(), ierr, NewTransformObject(), PREPROCESSOR, SysProDefineIntAnnotation(), TransformGetByName(), TransformObjectIntAnnotate(), and TransformObjectSetExplanation().

Referenced by DeclareScalingPreprocessor().

26.17.3.5 static PetscErrorCode specific_scaling_choices (NumericalProblem *theproblem*, SalsaTransform *scaling*) `[static]`

This is the 'specific setup' phase of the scaling preprocessor. See Usage modes for details.

This routine eliminates unsymmetric scalings if we are dealing with a symmetric system.

Definition at line 265 of file scaling.c.

References CHKERRQ(), ierr, SysProRetrieveQuantity(), TransformGetObjects(), TransformObjectGetIntAnnotation(), and TransformObjectMark().

Referenced by DeclareScalingPreprocessor().

26.17.3.6 static PetscErrorCode unscale_system (char ∗ *scaling_type*, PetscTruth *overwrite*, void ∗ *gctx*, void ∗ *ctx*, NumericalProblem *problem*, NumericalProblem *nextproblem*, NumericalSolution *scaled*, NumericalSolution *unscaled*) `[static]`

Definition at line 190 of file scaling.c.

References CHKERRQ(), DeleteLinearSystem(), ierr, LinearSolutionCopyStats(), and LinearSolutionGetVector().

Referenced by DeclareScalingPreprocessor().

26.18 singleton.c File Reference

```
#include <stdlib.h>
```

```
#include <stdio.h>
#include "petsc.h"
#include "syspro.h"
#include "sysprotransform.h"
#include "sysprolinear.h"
#include "linear_impl.h"
```

Data Structures

- struct singleton_struct

Defines

- #define PREPROCESSOR "singleton"

Functions

- static PetscErrorCode eliminate_singletons (char ∗type, int nopt, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode back_singleton (char ∗singleton_type, PetscTruth overwrite, void ∗gctx, void ∗ctx, NumericalProblem compactproblem, NumericalProblem fullproblem, NumericalSolution compactvector, NumericalSolution fullvector)
- static PetscErrorCode setup_singleton_choices ()
- static PetscErrorCode specific_singleton_choices (NumericalProblem theproblem, SalsaTransform singleton)
- static PetscErrorCode singleton_specific_unset (NumericalProblem theproblem)
- PetscErrorCode DeclareSingletonPreprocessor (void)

26.18.1 Detailed Description

Definition in file singleton.c.

26.18.2 Define Documentation

26.18.2.1 #define PREPROCESSOR "singleton"

Definition at line 19 of file singleton.c.

26.18.3 Function Documentation

26.18.3.1 static PetscErrorCode back_singleton (char ∗ *singleton_type*, PetscTruth *overwrite*, void ∗ *gctx*, void ∗ *ctx*, NumericalProblem *compactproblem*, NumericalProblem *fullproblem*, NumericalSolution *compactvector*, NumericalSolution *fullvector*) `[static]`

Definition at line 174 of file singleton.c.

References CHKERRQ(), DeleteLinearSystem(), singleton_struct::extractor, ierr, LinearSolutionCopyStats(), LinearSolutionGetVector(), LinearSolutionSetVector(), LinearSystemGetParts(), and singleton_struct::t.

Referenced by DeclareSingletonPreprocessor().

26.18.3.2 PetscErrorCode DeclareSingletonPreprocessor (void)

Definition at line 311 of file singleton.c.

References back_singleton(), CHKERRQ(), DeclarePreprocessor(), eliminate_singletons(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_singleton_choices(), singleton_specific_unset(), and specific_singleton_choices().

Referenced by main().

26.18.3.3 static PetscErrorCode eliminate_singletons (char ∗ *type*, int *nopt*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem ∗ *outproblem*, void ∗ *gctx*, void ∗∗ *ctx*, PetscTruth ∗ *success*) `[static]`

Definition at line 24 of file singleton.c.

References CHKERRQ(), singleton_struct::extractor, ierr, LinearSystemDuplicatePointers(), LinearSystemGetContext(), LinearSystemGetParts(), LinearSystemSetContext(), LinearSystemSetParts(), singleton_struct::n, SysProRetrieveQuantity(), and singleton_struct::t.

Referenced by DeclareSingletonPreprocessor().

26.18.3.4 static PetscErrorCode setup_singleton_choices () `[static]`

This routine is only called when the singleton preprocessor is created by DeclarePreprocessor() inside DeclareSingletonPreprocessor()

Definition at line 234 of file singleton.c.

References CHKERRQ(), ierr, NewTransformObject(), PREPROCESSOR, TransformGetByName(), and TransformObjectSetExplanation().

Referenced by DeclareSingletonPreprocessor().

26.18.3.5 static PetscErrorCode singleton_specific_unset (NumericalProblem *theproblem*) `[static]`

Definition at line 298 of file singleton.c.

References CHKERRQ(), ierr, and SysProRemoveQuantity().

Referenced by DeclareSingletonPreprocessor().

26.18.3.6 static PetscErrorCode specific_singleton_choices (NumericalProblem *theproblem*, SalsaTransform *singleton*) `[static]`

This is the 'specific setup' phase of the singleton preprocessor. See Usage modes for details.

It disables either the identity or the elimination routine, to leave only the one applicable to this particular system.

Maybe if we'd ever want to prove how effective singleton elimination is, we could leave identity in place for systems with singletons.

Definition at line 265 of file singleton.c.

References CHKERRQ(), ierr, PREPROCESSOR, SysProComputeQuantity(), TransformObjectGetByName(), and TransformObjectMark().

Referenced by DeclareSingletonPreprocessor().

26.19 suit.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "syspro.h"
#include "sysprolinear.h"
```

```
#include "linear_impl.h"
#include "sysprosuit.h"
#include "anamod.h"
#include "linksp.h"
#include "petscmat.h"
#include "petscpc.h"
#include "petscksp.h"
```

Functions

- PetscErrorCode onlyforsymmetricproblem (NumericalProblem problem, void ∗ctx, SuitabilityValue ∗v)

26.19.1 Function Documentation

26.19.1.1 PetscErrorCode onlyforsymmetricproblem (NumericalProblem *problem*, void ∗ *ctx*, SuitabilityValue ∗ *v*)

26.19.2 Suitability functions for the linear problem

Definition at line 18 of file suit.c.

References CHKERRQ(), and ierr.

Referenced by setup_ksp_choices().

26.20 syspro.h File Reference

```
#include "petscmat.h"
```

Defines

- #define TRUTH(x) ((x) ? PETSC_TRUE : PETSC_FALSE)

Typedefs

- typedef struct SystemPreprocessor_ ∗ SystemPreprocessor
- typedef struct NumericalProblem_ ∗ NumericalProblem

- typedef struct NumericalSolution_ ∗ NumericalSolution
- typedef PetscReal SuitabilityValue
- typedef struct SalsaTransform_ ∗ SalsaTransform
- typedef struct SalsaTransformObject_ ∗ SalsaTransformObject

Functions

- PetscErrorCode SysProInitialize ()
- PetscErrorCode SysProFinalize ()
- PetscErrorCode DeclarePreprocessor (char ∗name, PetscErrorCode(∗global_setup)(), PetscErrorCode(∗specific_-setup)(NumericalProblem, SalsaTransform), PetscErrorCode(∗specific_-unset)(NumericalProblem), PetscErrorCode(∗global_unset)(), PetscErrorCode(∗ctxcreate)(NumericalProblem, void ∗∗), PetscErrorCode(∗ctxdelete)(void ∗), PetscErrorCode(∗start_function)(char ∗, int, PetscTruth, NumericalProblem, NumericalProblem ∗, void ∗, void ∗∗, PetscTruth ∗), PetscErrorCode(∗end_function)(char ∗, PetscTruth, void ∗, void ∗, NumericalProblem, NumericalProblem, NumericalSolution, NumericalSolution))
- PetscErrorCode DeclarePreprocessorIntelligentChoice (char ∗name, PetscErrorCode(∗picker)(NumericalProblem, char ∗∗, char ∗∗))
- PetscErrorCode PreprocessorsOptionsHandling ()
- PetscErrorCode SysProDeclareFunctions (PetscErrorCode(∗classstaticsetup)(char ∗), PetscErrorCode(∗classdynamicsetup)(char ∗, NumericalProblem), PetscErrorCode(∗classproblemcloner)(char ∗, char ∗, int, NumericalProblem, NumericalProblem), PetscErrorCode(∗solver)(NumericalProblem, void ∗, NumericalSolution ∗), PetscErrorCode(∗problemdelete)(NumericalProblem), PetscErrorCode(∗solutioncreator)(NumericalProblem, NumericalSolution ∗), PetscErrorCode(∗solutioncopy)(NumericalSolution, NumericalSolution), PetscErrorCode(∗solutiondelete)(NumericalSolution), PetscErrorCode(∗ctxcloner)(char ∗, char ∗, void ∗, void ∗∗), PetscErrorCode(∗ctxfree)(void ∗), PetscErrorCode(∗solutioncontextdelete)(NumericalSolution))
- PetscErrorCode ProcessPreprocessorOptions (char ∗processor, void ∗ctx)
- PetscErrorCode PreprocessorGetIndex (char ∗, int ∗)
- PetscErrorCode SystemPreprocessorGetByName (char ∗, SystemPreprocessor ∗)
- PetscErrorCode PreprocessorGetSetting (char ∗, char ∗∗, int ∗)
- PetscErrorCode RetrievePreprocessorChoice (int, char ∗∗, int ∗)
- PetscErrorCode GetFirstPreprocessor (char ∗∗preprocess)
- PetscErrorCode GetNextPreprocessor (char ∗∗next_one)
- PetscErrorCode SuccessorPreprocessor (char ∗, char ∗∗)
- PetscErrorCode InitRetrievingPreprocessors ()

- PetscErrorCode StartRetrievingCurrentPreprocessors (char ∗∗, char ∗∗, int ∗, PetscTruth ∗)
- PetscErrorCode ContinueRetrievingCurrentPreprocessors (char ∗∗, char ∗∗, int ∗, PetscTruth ∗)
- PetscErrorCode StartRetrievingAllPreprocessors (char ∗∗, char ∗∗∗, int ∗, PetscTruth ∗)
- PetscErrorCode ContinueRetrievingAllPreprocessors (char ∗∗, char ∗∗∗, int ∗, PetscTruth ∗)
- PetscErrorCode RetrieveAllPreprocessorValues (char ∗, char ∗∗∗, int ∗)
- PetscErrorCode RegisterPreprocessorContext (char ∗pre, void ∗ctx)
- PetscErrorCode PreprocessorGetContext (char ∗pre, void ∗∗ctx)
- PetscErrorCode PreprocessedSolution (char ∗, NumericalProblem, void ∗, NumericalSolution ∗)
- PetscErrorCode PreprocessedProblemSolving (NumericalProblem, NumericalSolution ∗)
- PetscErrorCode PreprocessorSetPreservedCategories (char ∗, char ∗)
- PetscErrorCode PreprocessorGetPreservedCategories (char ∗, char ∗∗)
- PetscErrorCode SysProComputeQuantity (NumericalProblem, char ∗, char ∗, void ∗, int ∗, PetscTruth ∗)
- PetscErrorCode SysProRetrieveQuantity (NumericalProblem, char ∗, char ∗, void ∗, int ∗, PetscTruth ∗)
- PetscErrorCode SysProRemoveQuantity (NumericalProblem, char ∗, char ∗, PetscTruth ∗)
- PetscErrorCode SysProFreeQuantities (NumericalProblem)
- PetscErrorCode SysProDefaultTrace (void ∗ctx, char ∗fmt, va_list argp)
- PetscErrorCode SysProDeclareTraceFunction (PetscErrorCode(∗fn)(void ∗, char ∗, va_list))
- PetscErrorCode SysProDeclareTraceContext (void ∗ctx)
- PetscErrorCode SysProTraceMessage (char ∗fmt,...)
- PetscErrorCode SysProHasTrace (PetscTruth ∗flg)
- PetscErrorCode SysProDeclareProblemMonitor (PetscErrorCode(∗)(NumericalProblem))
- PetscErrorCode SysProDeclareErrorTracer (PetscErrorCode(∗)(NumericalProblem, NumericalSolution, char ∗))
- PetscErrorCode SysProGetErrorTracer (PetscErrorCode(∗∗)(NumericalProblem, NumericalSolution, char ∗))
- PetscErrorCode TabReportAllPreprocessors (char ∗∗key, int)
- PetscErrorCode TabReportActivePreprocessors (char ∗∗key, char ∗∗val, int)
- PetscErrorCode ReportEnabledPreprocessors (char ∗name)
- PetscErrorCode ReportSysProCallStackState (char ∗name)
- PetscErrorCode ScreenOutputTab (char ∗key, char ∗val)
- PetscErrorCode ScreenOutputTabLine (char ∗key, char ∗val)
- PetscErrorCode NumericalProblemGetComm (NumericalProblem, MPI_Comm ∗)

26.20.1 Define Documentation

26.20.1.1 #define TRUTH(x) ((x) ? PETSC_TRUE : PETSC_FALSE)

Definition at line 6 of file syspro.h.

Referenced by is_gmres_method(), PreprocessedSolution(), PreprocessorsOptionsHandling(), set_intelligent_scaling(), setup_pc(), TransformGetNextUnmarkedItem(), and TransformObjectGetIntAnnotation().

26.20.2 Typedef Documentation

26.20.2.1 typedef struct NumericalProblem_∗ NumericalProblem

Definition at line 13 of file syspro.h.

26.20.2.2 typedef struct NumericalSolution_∗ NumericalSolution

Definition at line 14 of file syspro.h.

26.20.2.3 typedef struct SalsaTransform_∗ SalsaTransform

Definition at line 17 of file syspro.h.

26.20.2.4 typedef struct SalsaTransformObject_∗ SalsaTransformObject

Definition at line 18 of file syspro.h.

26.20.2.5 typedef PetscReal SuitabilityValue

Definition at line 15 of file syspro.h.

26.20.2.6 typedef struct SystemPreprocessor_$*$ SystemPreprocessor

Definition at line 12 of file syspro.h.

26.20.3 Function Documentation

26.20.3.1 PetscErrorCode ContinueRetrievingAllPreprocessors (char $**$ *cclass*, char $***$ *types*, int $*$ *ntypes*, PetscTruth $*$ *success*)

This routine is to be used repeatedly after an initial call to StartRetrievingAllPreprocessors().

The `types` argument is allocated internally and should be deallocated by the user.

Definition at line 186 of file reporting.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, SystemPreprocessor_::name, npreprocess, preprocessreadout, SystemPreprocessor_::transform, and TransformObjectsGetNames().

Referenced by RetrieveAllPreprocessorValues(), StartRetrievingAllPreprocessors(), and TabReportPreprocessors().

26.20.3.2 PetscErrorCode ContinueRetrievingCurrentPreprocessors (char $**$ *cclass*, char $**$ *type*, int $*$ *opt*, PetscTruth $*$ *success*)

This routine is to be used repeatedly after an initial call to StartRetrievingCurrentPreprocessors().

Definition at line 125 of file reporting.c.

References CHKERRQ(), ierr, SystemPreprocessor_::name, preprocesslevel, preprocessreadout, and RetrievePreprocessorChoice().

Referenced by StartRetrievingCurrentPreprocessors(), and TabReportPreprocessors().

26.20.3.3 PetscErrorCode DeclarePreprocessor (char ∗ *name*, PetscErrorCode(∗)() *this_preprocessor_setup*, PetscErrorCode(∗)(NumericalProblem, SalsaTransform) *specific_setup*, PetscErrorCode(∗)(NumericalProblem) *specific_unset*, PetscErrorCode(∗)() *global_unset*, PetscErrorCode(∗)(NumericalProblem, void ∗∗) *ctxcreate*, PetscErrorCode(∗)(void ∗) *ctxdelete*, PetscErrorCode(∗)(char ∗, int, PetscTruth, NumericalProblem, NumericalProblem ∗, void ∗, void ∗∗, PetscTruth ∗) *start_function*, PetscErrorCode(∗)(char ∗, PetscTruth, void ∗, void ∗, NumericalProblem, NumericalProblem, NumericalSolution, NumericalSolution) *end_function*)

Declare a preprocessor class, by specifing its various members.

The `name` argument should not contain the colon character.

Here is an explanation of the various function arguments.

`this_preprocessor_setup()` : this routine is called only once, inside this function. This is a good place for defining all the preprocessors in this class

`specific_setup(NumericalProblem,SalsaTransform)` : this is called at the start of a preprocessing stage; one could use this for computing matrix metadata.

`global_unset(void)` : this is called in SysProFinalize().

`ctx_create(NumericalProblem,void**)` : create an object that can be used for the duration of the application of this preprocessor

`ctxdelete(void*)` : delete the context again

`start_function` : this is the function that performs the forward transform of the problem. Prototype:

```
PetscErrorCode start_function
  (char             *classmember,
   int               optionvalue,
   PetscTruth        overwrite,
   NumericalProblem  problem,
   NumericalProblem *transformedproblem,
   void             *globalcontext,
   void            **localcontext,
   PetscTruth       *success)
```

`end_function` : this is the backtransform. Its main task is copying or backtransforming the preprocessed solution to the original solution.

```
PetscErrorCode end_function
  (char             *classmember,
   PetscTruth        overwrite,
   void              *globalcontext,
   void              *localcontext,
   NumericalProblem  pproblem,
   NumericalProblem  oproblem,
   NumericalSolution psolution,
   NumericalSolution osolution)
```

where `pproblem` and `psolution` are the preprocessed quantities, the end function has to unprocess them and leave the result in `oproblem`, `osolution`. Actually, `oproblem` is only for reference.

Definition at line 319 of file preprocess.c.

References CHKERRQ(), PreprocessorsGlobalInfo_::classstaticsetup, SystemPreprocessor_::ctxcreate, SystemPreprocessor_::ctxdelete, SystemPreprocessor_::end_function, SystemPreprocessor_::exhaustive, ierr, SystemPreprocessor_::name, NewTransform(), NPREPROCESS, npreprocess, SystemPreprocessor_::setup, SystemPreprocessor_::start_function, SystemPreprocessor_::transform, SystemPreprocessor_::unset, and unsetpreprocessor.

Referenced by DeclareApproximationPreprocessor(), DeclareDistributionPreprocessor(), DeclareFlipsignPreprocessor(), DeclareKSPPreprocessor(), DeclarePCPreprocessor(), DeclareScalingPreprocessor(), DeclareSingletonPreprocessor(), and main().

26.20.3.4 PetscErrorCode DeclarePreprocessorIntelligentChoice (char ∗ *name*, PetscErrorCode(∗)(NumericalProblem, char ∗∗, char ∗∗) *picker*)

Install a function to pick the optimal choice for a preprocessor

Definition at line 399 of file preprocess.c.

References CHKERRQ(), ierr, SystemPreprocessor_::intelligence, and SystemPreprocessorGetByName().

Referenced by DeclareScalingPreprocessor().

26.20.3.5 PetscErrorCode GetFirstPreprocessor (char ∗∗ *preprocess*)

Get the name of the first declared preprocessor (in order of declaration) or null if none have been declared. Subsequent preprocessors can be retrieved with GetNextPreprocessor() or SuccessorPreprocessor().

Definition at line 45 of file reporting.c.

References CHKERRQ(), GetNextPreprocessor(), ierr, and preprocessreadout.

Referenced by PreprocessedProblemSolving(), and PreprocessorsOptionsHandling().

26.20.3.6 PetscErrorCode GetNextPreprocessor (char ∗∗ *next_one*)

Get the next preprocessor according to the variable preprocessreadout.

The result is null if there are no further preprocessors.

Definition at line 61 of file reporting.c.

References SystemPreprocessor_::name, npreprocess, and preprocessreadout.

Referenced by GetFirstPreprocessor(), and PreprocessorsOptionsHandling().

26.20.3.7 PetscErrorCode InitRetrievingPreprocessors ()

Definition at line 169 of file reporting.c.

References preprocessreadout.

Referenced by TabReportPreprocessors().

26.20.3.8 PetscErrorCode NumericalProblemGetComm (NumericalProblem, MPI_Comm ∗)

Definition at line 411 of file preprocess.c.

References NumericalProblem_::comm.

Referenced by create_solver().

26.20.3.9 PetscErrorCode PreprocessedProblemSolving (NumericalProblem *problem*, NumericalSolution ∗ *solution*)

Invoking this routine starts the preprocessing and ultimate solution of the numerical problem.

Definition at line 1063 of file preprocess.c.

References CHKERRQ(), PreprocessorsGlobalInfo_::errortracer, GetFirstPreprocessor(), ierr, PreprocessedSolution(), preprocesslevel, and PreprocessorsGlobalInfo_::problemsolver.

Referenced by main(), and PreprocessedLinearSystemSolution().

26.20.3.10 PetscErrorCode PreprocessedSolution (char ∗ *pclassname*, NumericalProblem *problem*, void ∗ *prevctx*, NumericalSolution ∗ *rsolution*)

This routine handles the application of one preprocessor. Depending on the runtime setup (see section Usage modes), one choice is applied, or a sequence of choices is applied consecutively. The forward and backward transformation of the preprocessor are done here, and if necessary, backup copies of the system are kept around.

Definition at line 868 of file preprocess.c.

References CHKERRQ(), ChooseFirstTransform(), PreprocessorsGlobalInfo_-::classdynamicsetup, SystemPreprocessor_::ctxcreate, SystemPreprocessor_-::ctxdelete, PreprocessorsGlobalInfo_::errortracer, SystemPreprocessor_::exhaustive, ierr, PreprocessedSolution(), preprocesslevel, PreprocessorSpecificSetup(), PreprocessorsGlobalInfo_::problemsolver, RegisterPreprocessorContext(), RegisterPreprocessorSetting(), ReportSysProCallStackState(), PreprocessorsGlobalInfo_-::solutioncontextdelete, PreprocessorsGlobalInfo_::solutiondelete, SuccessorPreprocessor(), SysProPreprocessorEndFunction(), SysproPreprocessorStartFunction(), SysProProblemCloneContext(), SysProTraceMessage(), SystemPreprocessorGetByName(), SystemPreprocessor_::transform, TransformGetNextUnmarkedItem(), TransformGetUserChoices(), TransformItemGetFirstOption(), TransformItemGetNextOption(), TransformObjectGetName(), TRUTH, and SystemPreprocessor_-::unset.

Referenced by PreprocessedProblemSolving(), and PreprocessedSolution().

26.20.3.11 PetscErrorCode PreprocessorGetContext (char ∗ *pre*, void ∗∗ *ctx*)

Definition at line 668 of file preprocess.c.

References CHKERRQ(), ierr, preprocessorcontexts, PreprocessorGetIndex(), and solutioncontext.

Referenced by setup_ksp(), and setup_pc().

26.20.3.12 PetscErrorCode PreprocessorGetIndex (char ∗, int ∗)

Definition at line 577 of file preprocess.c.

References CHKERRQ(), ierr, and npreprocess.

Referenced by PreprocessorGetContext(), RegisterPreprocessorContext(), and SystemPreprocessorGetByName().

26.20.3.13 PetscErrorCode PreprocessorGetPreservedCategories (char $*$, char $**$)

Definition at line 53 of file compute.c.

References CHKERRQ(), ierr, SystemPreprocessor_::preserved, and SystemPreprocessorGetByName().

26.20.3.14 PetscErrorCode PreprocessorGetSetting (char $*$, char $**$, int $*$)

Definition at line 545 of file preprocess.c.

References currentchoices, currentoptions, currentpreprocessors, and preprocesslevel.

Referenced by disable_ksps().

26.20.3.15 PetscErrorCode PreprocessorSetPreservedCategories (char $*$, char $*$)

Definition at line 32 of file compute.c.

References CHKERRQ(), ierr, SystemPreprocessor_::preserved, and SystemPreprocessorGetByName().

Referenced by DeclareApproximationPreprocessor(), DeclareDistributionPreprocessor(), DeclareFlipsignPreprocessor(), DeclareKSPPreprocessor(), DeclarePCPreprocessor(), DeclareScalingPreprocessor(), and DeclareSingletonPreprocessor().

26.20.3.16 PetscErrorCode PreprocessorsOptionsHandling ()

Process commandline options that control the behaviour of SysPro. For more information see Command line options handling.

Definition at line 46 of file options.c.

References CHKERRQ(), SystemPreprocessor_::exhaustive, GetFirstPreprocessor(), GetNextPreprocessor(), ierr, SystemPreprocessor_::optionshandling, Preprocessor-SaveAprioriSelection(), ReportEnabledPreprocessors(), SystemPreprocessorGetBy-Name(), TransformGetByName(), TransformGetNextUnmarkedItem(), TransformGet-NUnmarked(), TransformItemOptionsUseOnly(), TransformObjectGetName(), Trans-formObjectsUseOnly(), TransformSetUserChoices(), TRUTH, and TYPELEN.

Referenced by main().

26.20.3.17 PetscErrorCode ProcessPreprocessorOptions (char ∗ *processor*, void ∗ *ctx*)

26.20.3.18 PetscErrorCode RegisterPreprocessorContext (char ∗ *pre*, void ∗ *ctx*)

Definition at line 646 of file preprocess.c.

References CHKERRQ(), ierr, preprocessorcontexts, PreprocessorGetIndex(), and so-lutioncontext.

Referenced by PreprocessedLinearSystemSolution(), and PreprocessedSolution().

26.20.3.19 PetscErrorCode ReportEnabledPreprocessors (char ∗ *name*)

Report preprocessor choices that are available after the specific setup has possible dis-abled some of the registered ones. This function uses the `sysprotrace` function, so this has to have been declared.

Definition at line 387 of file reporting.c.

References CHKERRQ(), ierr, SysProHasTrace(), SysProTraceMessage(), Transfor-mGetByName(), and TransformReportEnabled().

Referenced by PreprocessorsOptionsHandling(), and PreprocessorSpecificSetup().

26.20.3.20 PetscErrorCode ReportSysProCallStackState (char ∗ *name*)

Report preprocessor choices that are available after the specific setup has possible dis-abled some of the registered ones. This function uses the `sysprotrace` function, so this has to have been declared.

Definition at line 411 of file reporting.c.

References CHKERRQ(), ierr, ScreenOutputTabLine(), SysProHasTrace(), SysProTraceMessage(), and TabReportPreprocessors().

Referenced by PreprocessedSolution().

26.20.3.21 PetscErrorCode RetrieveAllPreprocessorValues (char $*$, char $***$, int $*$)

Definition at line 210 of file reporting.c.

References CHKERRQ(), ContinueRetrievingAllPreprocessors(), ierr, npreprocess, and preprocessreadout.

Referenced by pcoptionshandling().

26.20.3.22 PetscErrorCode RetrievePreprocessorChoice (int, char $**$, int $*$)

Definition at line 565 of file preprocess.c.

References currentchoices, and currentoptions.

Referenced by ContinueRetrievingCurrentPreprocessors().

26.20.3.23 PetscErrorCode ScreenOutputTab (char $*$ *key*, char $*$ *val*)

Definition at line 320 of file reporting.c.

References CHKERRQ(), ierr, SysProHasTrace(), and SysProTraceMessage().

26.20.3.24 PetscErrorCode ScreenOutputTabLine (char $*$ *key*, char $*$ *val*)

Definition at line 347 of file reporting.c.

References CHKERRQ(), ierr, MAXLEN, SysProHasTrace(), and SysProTraceMessage().

Referenced by ReportSysProCallStackState().

26.20.3.25 PetscErrorCode StartRetrievingAllPreprocessors (char ∗∗ *cclass*, char ∗∗∗ *types*, int ∗ *ntypes*, PetscTruth ∗ *success*)

This routine gives the class of the first declared preprocessor, and all possible values. To get the next preprocessor, call ContinueRetrievingAllPreprocessors().

The class, types, and ntypes arguments can all be null.

The `types` argument is allocated internally and should be deallocated by the user.

Definition at line 157 of file reporting.c.

References CHKERRQ(), ContinueRetrievingAllPreprocessors(), ierr, and preprocess-readout.

26.20.3.26 PetscErrorCode StartRetrievingCurrentPreprocessors (char ∗∗ *cclass*, char ∗∗ *type*, int ∗ *opt*, PetscTruth ∗ *success*)

This routine gives the class and current value of the first declared preprocessor. To get the next preprocessor, call ContinueRetrievingAllPreprocessors().

The class, types, and ntypes arguments can all be null.

Definition at line 108 of file reporting.c.

References CHKERRQ(), ContinueRetrievingCurrentPreprocessors(), ierr, and prepro-cessreadout.

26.20.3.27 PetscErrorCode SuccessorPreprocessor (char ∗ *this_one*, char ∗∗ *next_one*)

Given a preprocessor, get the name of the next one (in order of declaration) or null if there are no further ones.

The arguments are allowed to be the same.

Definition at line 78 of file reporting.c.

References CHKERRQ(), ierr, SystemPreprocessor_::name, and npreprocess.

Referenced by PreprocessedSolution().

26.20.3.28 PetscErrorCode SysProComputeQuantity (NumericalProblem *theproblem*, char ∗ *cat*, char ∗ *cmp*, void ∗ *res*, int ∗ *reslen*, PetscTruth ∗ *flg*)

anamod SysPro-AnaMod interface

The SysPro linear package has a few routines to facilitate integration with AnaMod

- `SysProComputeQuantity()` : to compute a quantity using AnaMod and store it as the metadata of a linear system
- `SysProRetrieveQuantity()` : to get an already computed quantity
- `SysProFreeQuantities()` : to destroy the metadata object
- `SysProRemoveQuantity()` : to invalidate/free selected quantities

This routine is used in SysPro to compute quantities. See also SysProRetrieveQuantity().

Definition at line 23 of file syspro_anamod.c.

References CHKERRQ(), ierr, LinearSystemGetMetadata(), and LinearSystemGetParts().

Referenced by flipsign(), MatSymmetricPart(), sans_partition(), specific_flipsign_-choices(), and specific_singleton_choices().

26.20.3.29 PetscErrorCode SysProDeclareErrorTracer (PetscErrorCode(∗)(NumericalProblem, NumericalSolution, char ∗))

Definition at line 378 of file preprocess.c.

References PreprocessorsGlobalInfo_::errortracer.

26.20.3.30 PetscErrorCode SysProDeclareFunctions (PetscErrorCode(∗)(char ∗) *classstaticsetup*, PetscErrorCode(∗)(char ∗, NumericalProblem) *classdynamicsetup*, PetscErrorCode(∗)(char ∗, char ∗, int, NumericalProblem, NumericalProblem) *classproblemcloner*, PetscErrorCode(∗)(NumericalProblem, void ∗, NumericalSolution ∗) *problemsolver*, PetscErrorCode(∗)(NumericalProblem) *problemdelete*, PetscErrorCode(∗)(NumericalProblem, NumericalSolution ∗) *solutioncreator*, PetscErrorCode(∗)(NumericalSolution, NumericalSolution) *solutioncopy*, PetscErrorCode(∗)(NumericalSolution) *solutiondelete*, PetscErrorCode(∗)(char ∗, char ∗, void ∗, void ∗∗) *ctxcloner*, PetscErrorCode(∗)(void ∗) *ctxfree*, PetscErrorCode(∗)(NumericalSolution) *solutioncontextdelete*)

Install various functions

- `classstaticsetup` : this function is called on each processor as it is being created; see DeclarePreprocessor().
- `classdynamicsetup` : this function is called as any invocation of a preprocessor starts; see PreprocessedSolution();
- `classproblemcloner` : a function to clone the context : optional see Tracing the preprocessors for more details.
- `problemsolver` : the ultimate problem solver : required
- `problemdelete` : delete a problem object
- `solutioncreator` : creates a solution object; optional, but required a preprocessor has an endfunction.
- `solutioncopy` : guess what this does; optional
- `solutiondelete` : optional, but needed if solutioncopy is used
- `contextcloner` : problems can carry a context; this clones the context if a problem is copied; otherwise the pointer is simply duplicated
- `contextfree` : used to delete cloned contexts
- `solutioncontextdelete` : hm.

Definition at line 440 of file preprocess.c.

References PreprocessorsGlobalInfo_::classdynamicsetup, PreprocessorsGlobalInfo_::classproblemcloner, PreprocessorsGlobalInfo_::classstaticsetup, PreprocessorsGlobalInfo_::clonecontext, PreprocessorsGlobalInfo_::freecontext, PreprocessorsGlobalInfo_::problemdelete, PreprocessorsGlobalInfo_-::problemsolver, PreprocessorsGlobalInfo_::solutioncontextdelete, PreprocessorsGlobalInfo_::solutioncopy, PreprocessorsGlobalInfo_::solutioncreator, and PreprocessorsGlobalInfo_::solutiondelete.

Referenced by main().

26.20.3.31 PetscErrorCode SysProDeclareProblemMonitor (PetscErrorCode(∗)(NumericalProblem))

Definition at line 368 of file preprocess.c.

References PreprocessorsGlobalInfo_::problemmonitor.

26.20.3.32 PetscErrorCode SysProDeclareTraceContext (void ∗ *ctx*)

Definition at line 77 of file tracing.c.

References sysprotracectx.

26.20.3.33 PetscErrorCode SysProDeclareTraceFunction (PetscErrorCode(∗)(void ∗, char ∗, va_list) *fn*)

Specify a trace function.

The trace function has a prototype

```
  PetscErrorCode tracefunction(void*,char*,va_list)
```

which means that it has an arbitrary number of arguments, much like `printf`. The first argument is a context, which can be set by SysProDeclareTraceContext().

Here is an example of how you would write a trace function:

```
#include <stdarg.h>
PetscErrorCode tracefunction(void *ctx,char *fmt,va_list argp)
{
  char *prefix = (char*)ctx;
  PetscFunctionBegin;
  printf("%s ",prefix);
  vprintf(fmt, argp);
  PetscFunctionReturn(0);
}
```

Consult `string.h` (probably in `/usr/include`) to see which "v" versions of `printf` are available.

There is a default trace function SysProDefaultTrace().

You can undeclare a trace function by passing NULL.

See also SysProTraceMessage().

Definition at line 64 of file tracing.c.

References sysprotrace.

Referenced by main().

26.20.3.34 PetscErrorCode SysProDefaultTrace (void ∗ *ctx*, char ∗ *fmt*, va_list *argp*)

Definition at line 22 of file tracing.c.

Referenced by main().

26.20.3.35 PetscErrorCode SysProFinalize ()

Definition at line 236 of file preprocess.c.

References CHKERRQ(), currentchoices, currentoptions, currentpreprocessors, DeregisterTransform(), ierr, SystemPreprocessor_::name, npreprocess, preprocessorcontexts, SystemPreprocessor_::preserved, SystemPreprocessor_::transform, and unsetpreprocessor.

Referenced by main().

26.20.3.36 PetscErrorCode SysProFreeQuantities (NumericalProblem)

Definition at line 96 of file syspro_anamod.c.

References CHKERRQ(), ierr, and LinearSystemGetMetadata().

26.20.3.37 PetscErrorCode SysProGetErrorTracer (PetscErrorCode(∗∗)(NumericalProblem, NumericalSolution, char ∗))

Definition at line 388 of file preprocess.c.

References PreprocessorsGlobalInfo_::errortracer.

26.20.3.38 PetscErrorCode SysProHasTrace (PetscTruth ∗ *flg*)

Test whether a trace function has been declared; see SysProDeclareTraceFunction(). Normally you would use SysProTraceMessage() which performs this test internally, but this function can be useful if a large amount of processing has to be performed to construct the trace message to begin with.

Definition at line 109 of file tracing.c.

References sysprotrace.

Referenced by ReportEnabledPreprocessors(), ReportSysProCallStackState(), ScreenOutputTab(), and ScreenOutputTabLine().

26.20.3.39 PetscErrorCode SysProInitialize ()

Allocate SysPro globals. See also SysProFinalize().

Definition at line 211 of file preprocess.c.

References CHKERRQ(), CreateGlobalInfo(), currentchoices, currentoptions, currentpreprocessors, ierr, NPREPROCESS, preprocessorcontexts, and unsetpreprocessor.

Referenced by main().

26.20.3.40 PetscErrorCode SysProRemoveQuantity (NumericalProblem *theproblem*, char ∗ *cat*, char ∗ *cmp*, PetscTruth ∗ *flg*)

This routine is used to invalidate and free computed quantities. See also SysProRetrieveQuantity(), SysProComputeQuantity().

Definition at line 80 of file syspro_anamod.c.

References CHKERRQ(), ierr, and LinearSystemGetMetadata().

Referenced by singleton_specific_unset().

26.20.3.41 PetscErrorCode SysProRetrieveQuantity (NumericalProblem *theproblem*, char ∗ *cat*, char ∗ *cmp*, void ∗ *res*, int ∗ *reslen*, PetscTruth ∗ *flg*)

This routine is used in SysPro to retrieve already computed quantities. Reports failure if the quantity has not already been computed. See also SysProComputeQuantity().

Definition at line 52 of file syspro_anamod.c.

References CHKERRQ(), ierr, and LinearSystemGetParts().

Referenced by disable_pcs(), eliminate_singletons(), MatSymmetricPart(), set_-intelligent_scaling(), specific_approximation_choices(), and specific_scaling_-choices().

26.20.3.42 PetscErrorCode SysProTraceMessage (char ∗ *fmt*, ...)

This function prints a trace message if a trace function has been declared; see SysProDeclareTraceFunction().

Definition at line 89 of file tracing.c.

References CHKERRQ(), ierr, sysprotrace, and sysprotracectx.

Referenced by adder(), ChooseFirstTransform(), PreprocessedSolution(), ReportEnabledPreprocessors(), ReportSysProCallStackState(), ScreenOutputTab(), ScreenOutputTabLine(), and solvebycopy().

26.20.3.43 PetscErrorCode SystemPreprocessorGetByName (char ∗, SystemPreprocessor ∗)

Definition at line 598 of file preprocess.c.

References CHKERRQ(), ierr, and PreprocessorGetIndex().

Referenced by DeclarePCPreprocessor(), DeclarePreprocessorIntelligentChoice(), DeclarePreprocessorRequiredCategories(), PreprocessedSolution(), PreprocessorGetPreservedCategories(), PreprocessorSetPreservedCategories(), PreprocessorsOptionsHandling(), PreprocessorSpecificSetup(), and TransformGetByName().

26.20.3.44 PetscErrorCode TabReportActivePreprocessors (char ∗∗ *key*, char ∗∗ *val*, int)

Definition at line 309 of file reporting.c.

References CHKERRQ(), ierr, and TabReportPreprocessors().

26.20.3.45 PetscErrorCode TabReportAllPreprocessors (char ∗∗ *key*, int)

Definition at line 295 of file reporting.c.

References CHKERRQ(), ierr, and TabReportPreprocessors().

26.21 syspro_anamod.c File Reference

```
#include <stdlib.h>
#include "anamod.h"
#include "syspro.h"
#include "sysprolinear.h"
```

Functions

- PetscErrorCode SysProComputeQuantity (NumericalProblem theproblem, char ∗cat, char ∗cmp, void ∗res, int ∗reslen, PetscTruth ∗flg)
- PetscErrorCode SysProRetrieveQuantity (NumericalProblem theproblem, char ∗cat, char ∗cmp, void ∗res, int ∗reslen, PetscTruth ∗flg)
- PetscErrorCode SysProRemoveQuantity (NumericalProblem theproblem, char ∗cat, char ∗cmp, PetscTruth ∗flg)
- PetscErrorCode SysProFreeQuantities (NumericalProblem theproblem)

26.21.1 Function Documentation

26.21.1.1 PetscErrorCode SysProComputeQuantity (NumericalProblem *theproblem*, char ∗ *cat*, char ∗ *cmp*, void ∗ *res*, int ∗ *reslen*, PetscTruth ∗ *flg*)

anamod SysPro-AnaMod interface

The SysPro linear package has a few routines to facilitate integration with AnaMod

- `SysProComputeQuantity()` : to compute a quantity using AnaMod and store it as the metadata of a linear system
- `SysProRetrieveQuantity()` : to get an already computed quantity
- `SysProFreeQuantities()` : to destroy the metadata object
- `SysProRemoveQuantity()` : to invalidate/free selected quantities

This routine is used in SysPro to compute quantities. See also SysProRetrieveQuantity().

Definition at line 23 of file syspro_anamod.c.

References CHKERRQ(), ierr, LinearSystemGetMetadata(), and LinearSystemGetParts().

Referenced by flipsign(), MatSymmetricPart(), sans_partition(), specific_flipsign_-choices(), and specific_singleton_choices().

26.21.1.2 PetscErrorCode SysProFreeQuantities (NumericalProblem *theproblem*)

Definition at line 96 of file syspro_anamod.c.

References CHKERRQ(), ierr, and LinearSystemGetMetadata().

26.21.1.3 PetscErrorCode SysProRemoveQuantity (NumericalProblem *theproblem*, char ∗ *cat*, char ∗ *cmp*, PetscTruth ∗ *flg*)

This routine is used to invalidate and free computed quantities. See also SysProRetrieveQuantity(), SysProComputeQuantity().

Definition at line 80 of file syspro_anamod.c.

References CHKERRQ(), ierr, and LinearSystemGetMetadata().

Referenced by singleton_specific_unset().

26.21.1.4 PetscErrorCode SysProRetrieveQuantity (NumericalProblem *theproblem*, char ∗ *cat*, char ∗ *cmp*, void ∗ *res*, int ∗ *reslen*, PetscTruth ∗ *flg*)

This routine is used in SysPro to retrieve already computed quantities. Reports failure if the quantity has not already been computed. See also SysProComputeQuantity().

Definition at line 52 of file syspro_anamod.c.

References CHKERRQ(), ierr, and LinearSystemGetParts().

Referenced by disable_pcs(), eliminate_singletons(), MatSymmetricPart(), set_-intelligent_scaling(), specific_approximation_choices(), and specific_scaling_-choices().

26.22 syspro_impl.h File Reference

```
#include "petsc.h"
#include "syspro.h"
```

Data Structures

- struct NumericalProblem_
- struct SystemPreprocessor_
- struct SalsaTransformObject_
- struct SalsaTransform_

Defines

- #define SYSPROCHECKVALID(i, c, s) {if (!i) SETERRQ1(1,"Null pointer for <%s>",s); if (i → cookie!=c) SETERRQ1(1,"Not a valid <%s>",s);}
- #define SYSPROCHECKVALIDa(i, c, s, a) {if (!i) SETERRQ2(1,"Null pointer for <%s>, argument %d",s,a); if (i → cookie!=c) SETERRQ2(1,"Not a valid <%s>, argument %d",s,a);}
- #define NUMERICALPROBLEMHEADER MPI_Comm comm; void ∗ctx;

26.22.1 Define Documentation

26.22.1.1 #define NUMERICALPROBLEMHEADER MPI_Comm comm; void ∗ctx;

Definition at line 10 of file syspro_impl.h.

26.22.1.2 #define SYSPROCHECKVALID(i, c, s) {if (!i) SETERRQ1(1,"Null pointer for <%s>",s); if (i → cookie!=c) SETERRQ1(1,"Not a valid <%s>",s);}

Definition at line 7 of file syspro_impl.h.

26.22.1.3 #define SYSPROCHECKVALIDa(i, c, s, a) {if (!i) SETERRQ2(1,"Null pointer for <%s>, argument %d",s,a); if (i → cookie!=c) SETERRQ2(1,"Not a valid <%s>, argument %d",s,a);}

Definition at line 8 of file syspro_impl.h.

26.23 sysprolinear.h File Reference

```
#include "petscmat.h"
#include "petscksp.h"
#include "syspro.h"
#include "nmd.h"
```

Typedefs

- typedef struct LinearSystem_ ∗ LinearSystem
- typedef struct LinearSolution_ ∗ LinearSolution
- typedef struct Diagnostics_ ∗ Diagnostics

Functions

- PetscErrorCode CreateLinearSystem (MPI_Comm, LinearSystem ∗)
- PetscErrorCode DeleteLinearSystem (LinearSystem)
- PetscErrorCode LinearSystemSetParts (LinearSystem, Mat, Mat, Vec, Vec, Vec)
- PetscErrorCode LinearSystemInheritParts (LinearSystem, Mat, Mat, Vec, Vec, Vec)
- PetscErrorCode LinearSystemGetParts (LinearSystem, Mat ∗, Mat ∗, Vec ∗, Vec ∗, Vec ∗)
- PetscErrorCode LinearSystemGetTmpVector (LinearSystem, Vec ∗)
- PetscErrorCode LinearSystemSetContext (LinearSystem, void ∗ctx)
- PetscErrorCode LinearSystemGetContext (LinearSystem, void ∗∗ctx)
- PetscErrorCode LinearSystemSetKnownSolution (LinearSystem, PetscTruth)

- PetscErrorCode LinearSystemGetKnownSolution (LinearSystem, PetscTruth ∗)
- PetscErrorCode LinearSystemSetMetadata (LinearSystem, NMD_metadata)
- PetscErrorCode LinearSystemGetMetadata (LinearSystem, NMD_metadata ∗)
- PetscErrorCode LinearSystemDuplicatePointers (LinearSystem, LinearSystem ∗)
- PetscErrorCode LinearSystemDuplicate (LinearSystem, LinearSystem ∗)
- PetscErrorCode LinearSystemCopy (LinearSystem, LinearSystem)
- PetscErrorCode CreateLinearSolution (LinearSolution ∗)
- PetscErrorCode LinearCreateNumericalSolution (NumericalProblem, NumericalSolution ∗)
- PetscErrorCode LinearSolutionDelete (LinearSolution)
- PetscErrorCode LinearSolutionCopy (LinearSolution, LinearSolution)
- PetscErrorCode LinearSolutionCopyStats (LinearSolution, LinearSolution)
- PetscErrorCode LinearCopyNumericalSolution (NumericalSolution, NumericalSolution)
- PetscErrorCode LinearDeleteNumericalSolution (NumericalSolution)
- PetscErrorCode LinearDeleteNumericalSolutionContext (NumericalSolution)
- PetscErrorCode CreateDefaultLinearSolution (NumericalProblem, NumericalSolution ∗)
- PetscErrorCode LinearSolutionSetVector (LinearSolution, Vec)
- PetscErrorCode LinearSolutionGetVector (LinearSolution, Vec ∗)
- PetscErrorCode LinearSolutionGetStatistics (LinearSolution, NMD_metadata ∗)
- PetscErrorCode LinearSolutionSetTimes (LinearSolution, PetscLogDouble, PetscLogDouble, PetscLogDouble)
- PetscErrorCode LinearSolutionAddToPreprocessTime (LinearSolution, PetscLogDouble)
- PetscErrorCode LinearSolutionGetTimes (LinearSolution, PetscLogDouble ∗, PetscLogDouble ∗, PetscLogDouble ∗)
- PetscErrorCode LinearSolutionSetContext (LinearSolution, void ∗ctx)
- PetscErrorCode LinearSolutionGetContext (LinearSolution, void ∗∗ctx)
- PetscErrorCode LinearSystemTrueDistance (LinearSystem, LinearSolution, PetscReal ∗)
- PetscErrorCode LinearSystemTrueDistancePrint (NumericalProblem, NumericalSolution, char ∗)
- PetscErrorCode LinearSolutionCreateStatistics (LinearSolution sol)
- PetscErrorCode PreprocessedLinearSystemSolution (LinearSystem, LinearSolution ∗)
- PetscErrorCode delete_diagnostics (Diagnostics)
- PetscErrorCode make_diagnostics (char ∗, char ∗, Diagnostics ∗)
- PetscErrorCode DeclareSingletonPreprocessor (void)
- PetscErrorCode DeclareFlipsignPreprocessor (void)
- PetscErrorCode DeclareApproximationPreprocessor (void)

- PetscErrorCode DeclareDummyRowPreprocessor (void)
- PetscErrorCode DeclareDistributionPreprocessor (void)
- PetscErrorCode DeclareScalingPreprocessor (void)
- PetscErrorCode DeclarePCPreprocessor (void)
- PetscErrorCode DeclareKSPPreprocessor (void)

26.23.1 Typedef Documentation

26.23.1.1 typedef struct Diagnostics_∗ Diagnostics

Definition at line 11 of file sysprolinear.h.

26.23.1.2 typedef struct LinearSolution_∗ LinearSolution

Definition at line 10 of file sysprolinear.h.

26.23.1.3 typedef struct LinearSystem_∗ LinearSystem

Definition at line 9 of file sysprolinear.h.

26.23.2 Function Documentation

26.23.2.1 PetscErrorCode CreateDefaultLinearSolution (NumericalProblem, NumericalSolution ∗)

Definition at line 516 of file linear.c.

References CHKERRQ(), CreateLinearSolution(), ierr, LinearSolutionSetVector(), LinearSystemGetParts(), and SYSPROCHECKVALIDLINSYS.

26.23.2.2 PetscErrorCode CreateLinearSolution (LinearSolution ∗)

Definition at line 399 of file linear.c.

References CHKERRQ(), LinearSolution_::cookie, ierr, LINSOLCOOKIE, and LinearSolution_::statistics.

Referenced by CreateDefaultLinearSolution(), and LinearCreateNumericalSolution().

26.23.2.3 PetscErrorCode CreateLinearSystem (MPI_Comm *comm*, LinearSystem $*$ *system*)

Allocate the structure for a linear system

Definition at line 75 of file linear.c.

References CHKERRQ(), LinearSystem_::cookie, ierr, LINSYSCOOKIE, and LinearSystem_::partsoriginal.

Referenced by LinearSystemDuplicate(), LinearSystemDuplicatePointers(), and main().

26.23.2.4 PetscErrorCode DeclareApproximationPreprocessor (void)

Definition at line 280 of file approximating.c.

References approximate_system(), CHKERRQ(), DeclarePreprocessor(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_approximation_-choices(), specific_approximation_choices(), and unapproximate_system().

26.23.2.5 PetscErrorCode DeclareDistributionPreprocessor (void)

Definition at line 299 of file distribution.c.

References CHKERRQ(), DeclarePreprocessor(), distribute_system(), ierr, PRE-PROCESSOR, PreprocessorSetPreservedCategories(), setup_distribution_choices(), specific_distribution_choices(), and undistribute_system().

26.23.2.6 PetscErrorCode DeclareDummyRowPreprocessor (void)

26.23.2.7 PetscErrorCode DeclareFlipsignPreprocessor (void)

Definition at line 168 of file flipsign.c.

References back_flipsign(), CHKERRQ(), DeclarePreprocessor(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_flipsign_choices(), and specific_flipsign_choices().

26.23.2.8 PetscErrorCode DeclareKSPPreprocessor (void)

Definition at line 367 of file ksp.c.

References CHKERRQ(), DeclarePreprocessor(), disable_ksps(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_ksp(), setup_ksp_choices(), unset_ksp(), and unset_ksps().

26.23.2.9 PetscErrorCode DeclarePCPreprocessor (void)

Definition at line 392 of file pc.c.

References CHKERRQ(), create_solver(), DeclarePreprocessor(), destroy_solver(), disable_pcs(), ierr, SystemPreprocessor_::optionshandling, pcoptionshandling(), PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_pc(), setup_pc_choices(), SystemPreprocessorGetByName(), and unset_pc().

26.23.2.10 PetscErrorCode DeclareScalingPreprocessor (void)

Definition at line 292 of file scaling.c.

References CHKERRQ(), DeclarePreprocessor(), DeclarePreprocessorIntelligentChoice(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), scale_system(), set_intelligent_scaling(), setup_scaling_choices(), specific_scaling_choices(), and unscale_system().

Referenced by main().

26.23.2.11 PetscErrorCode DeclareSingletonPreprocessor (void)

Definition at line 311 of file singleton.c.

References back_singleton(), CHKERRQ(), DeclarePreprocessor(), eliminate_singletons(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup_singleton_choices(), singleton_specific_unset(), and specific_singleton_choices().

Referenced by main().

26.23.2.12 PetscErrorCode delete_diagnostics (Diagnostics)

26.23.2.13 PetscErrorCode DeleteLinearSystem (LinearSystem)

Definition at line 90 of file linear.c.

References LinearSystem_::A, LinearSystem_::B, CHKERRQ(), ierr, LinearSystem_-::Init, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, SYSPROCHECKVALIDLINSYS, and LinearSystem_::Tmp.

Referenced by back_flipsign(), back_singleton(), unapproximate_system(), undistribute_system(), unscale_system(), unset_ksp(), and unset_pc().

26.23.2.14 PetscErrorCode LinearCopyNumericalSolution (NumericalSolution *old*, NumericalSolution *nnew*)

This routine is essentially LinearSolutionCopy(), except that it does casts of the arguments so that it can be used as the `solutioncopy` member of SysProDeclareFunctions()

Definition at line 503 of file linear.c.

References CHKERRQ(), ierr, and LinearSolutionCopy().

26.23.2.15 PetscErrorCode LinearCreateNumericalSolution (NumericalProblem *prob*, NumericalSolution ∗ *sol*)

Shell routine around CreateLinearSolution() to save you some type casting.

If the first argument is not NULL, its matrix is extracted and used to create the vector of the solution object.

Definition at line 420 of file linear.c.

References CHKERRQ(), CreateLinearSolution(), ierr, LinearSolutionSetVector(), and LinearSystemGetParts().

Referenced by main(), and solvelinear().

26.23.2.16 PetscErrorCode LinearDeleteNumericalSolution (NumericalSolution *sol*)

This is like LinearSolutionDelete(), except that the argument has been cast so that this routine can be used as the `solutiondelete` argument of SysProDeclareFunctions().

Definition at line 465 of file linear.c.

References CHKERRQ(), ierr, and LinearSolutionDelete().

Referenced by main().

26.23.2.17 PetscErrorCode LinearDeleteNumericalSolutionContext (NumericalSolution)

Definition at line 638 of file linear.c.

References LinearSolution_::ctx, and SYSPROCHECKVALIDLINSOL.

26.23.2.18 PetscErrorCode LinearSolutionAddToPreprocessTime (LinearSolution, PetscLogDouble)

26.23.2.19 PetscErrorCode LinearSolutionCopy (LinearSolution *old*, LinearSolution *lnew*)

Copy one linear solution object into another. This clearly only works if their vectors are similarly layed out.

The context pointer is blindly copied. We may have to think about this a bit more.

See also LinearCopyNumericalSolution().

Definition at line 484 of file linear.c.

References CHKERRQ(), LinearSolution_::ctx, ierr, LinearSolution_::Out, LinearSolution_::statistics, and SYSPROCHECKVALIDLINSOLa.

Referenced by back_flipsign(), LinearCopyNumericalSolution(), unapproximate_-system(), unset_ksp(), and unset_pc().

26.23.2.20 PetscErrorCode LinearSolutionCopyStats (LinearSolution, LinearSolution)

Definition at line 599 of file linear.c.

References CHKERRQ(), ierr, LinearSolution_::statistics, and SYSPROCHECK-VALIDLINSOLa.

Referenced by back_singleton(), undistribute_system(), and unscale_system().

26.23.2.21 PetscErrorCode LinearSolutionCreateStatistics (LinearSolution *sol*)

Definition at line 561 of file linear.c.

References CHKERRQ(), ierr, LinearSolution_::statistics, and SYSPROCHECK-VALIDLINSOL.

26.23.2.22 PetscErrorCode LinearSolutionDelete (LinearSolution *sol*)

Delete a linear solution.

This does not affect the context stored in the solution. That needs a special purpose routine.

See also LinearDeleteNumericalSolution().

Definition at line 447 of file linear.c.

References CHKERRQ(), ierr, LinearSolution_::Out, LinearSolution_::statistics, and SYSPROCHECKVALIDLINSOL.

Referenced by LinearDeleteNumericalSolution().

26.23.2.23 PetscErrorCode LinearSolutionGetContext (LinearSolution, void ** *ctx*)

Definition at line 628 of file linear.c.

References LinearSolution_::ctx, and SYSPROCHECKVALIDLINSOL.

26.23.2.24 PetscErrorCode LinearSolutionGetStatistics (LinearSolution, NMD_metadata ∗)

Definition at line 589 of file linear.c.

References LinearSolution_::statistics, and SYSPROCHECKVALIDLINSOL.

26.23.2.25 PetscErrorCode LinearSolutionGetTimes (LinearSolution, PetscLogDouble ∗, PetscLogDouble ∗, PetscLogDouble ∗)

26.23.2.26 PetscErrorCode LinearSolutionGetVector (LinearSolution, Vec ∗)

Definition at line 545 of file linear.c.

References LinearSolution_::Out, and SYSPROCHECKVALIDLINSOL.

Referenced by back_singleton(), LinearSystemTrueDistance(), LinearSystemTrueDistancePrint(), main(), undistribute_system(), and unscale_system().

26.23.2.27 PetscErrorCode LinearSolutionSetContext (LinearSolution, void ∗ *ctx*)

Definition at line 618 of file linear.c.

References LinearSolution_::ctx, and SYSPROCHECKVALIDLINSOL.

26.23.2.28 PetscErrorCode LinearSolutionSetTimes (LinearSolution, PetscLogDouble, PetscLogDouble, PetscLogDouble)

26.23.2.29 PetscErrorCode LinearSolutionSetVector (LinearSolution, Vec)

Definition at line 535 of file linear.c.

References LinearSolution_::Out, and SYSPROCHECKVALIDLINSOL.

Referenced by back_singleton(), CreateDefaultLinearSolution(), LinearCreateNumericalSolution(), and solvelinear().

26.23.2.30 PetscErrorCode LinearSystemCopy (LinearSystem *old*, LinearSystem *lnew*)

Copy the values of the components of an old linear system into a new. The new system has to have been created with LinearSystemDuplicate() because this routine assumes that the data structures are already in place.

Definition at line 356 of file linear.c.

References LinearSystem_::A, ALLPARTSNEW, LinearSystem_::B, CHKERRQ(), ierr, LinearSystem_::Init, LinearSystem_::known_solution, LinearSystem_::metadata, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYSa.

Referenced by scale_system().

26.23.2.31 PetscErrorCode LinearSystemDuplicate (LinearSystem *problem*, LinearSystem ∗ *newproblem*)

Allocate a new linear system, and create copies in it of the data structure, but not the values, of the components of the old system.

See also LinearSystemCopy().

Definition at line 311 of file linear.c.

References LinearSystem_::A, ALLPARTSNEW, LinearSystem_::B, CHKERRQ(), CreateLinearSystem(), NumericalProblem_::ctx, LinearSystem_::ctx, ierr, LinearSystem_::Init, LinearSystem_::known_solution, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYS.

Referenced by scale_system().

26.23.2.32 PetscErrorCode LinearSystemDuplicatePointers (LinearSystem *problem*, LinearSystem ∗ *newproblem*)

Allocate a new linear system and give it the components of the old by pointer duplication.

Definition at line 282 of file linear.c.

References LinearSystem_::A, LinearSystem_::B, CHKERRQ(), CreateLinearSystem(), NumericalProblem_::ctx, LinearSystem_::ctx, ierr, LinearSystem_::Init, LinearSystem_::known_solution, LinearSystem_::metadata, LinearSystem_-::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECK-VALIDLINSYS.

Referenced by approximate_system(), distribute_system(), eliminate_singletons(), flipsign(), setup_ksp(), and setup_pc().

26.23.2.33 PetscErrorCode LinearSystemGetContext (LinearSystem, void ∗∗ *ctx*)

Definition at line 215 of file linear.c.

References LinearSystem_::ctx, and SYSPROCHECKVALIDLINSYS.

Referenced by eliminate_singletons().

26.23.2.34 PetscErrorCode LinearSystemGetKnownSolution (LinearSystem, PetscTruth ∗)

Definition at line 235 of file linear.c.

References LinearSystem_::known_solution, and SYSPROCHECKVALIDLINSYS.

Referenced by LinearSystemTrueDistancePrint().

26.23.2.35 PetscErrorCode LinearSystemGetMetadata (LinearSystem, NMD_metadata ∗)

Definition at line 255 of file linear.c.

References LinearSystem_::metadata, and SYSPROCHECKVALIDLINSYS.

Referenced by main(), SysProComputeQuantity(), SysProFreeQuantities(), and SysProRemoveQuantity().

26.23.2.36 PetscErrorCode LinearSystemGetParts (LinearSystem *system*, Mat ∗ *A*, Mat ∗ *B*, Vec ∗ *Rhs*, Vec ∗ *Sol*, Vec ∗ *Init*)

Get the matrices and vectors of the system

Definition at line 190 of file linear.c.

References LinearSystem_::A, LinearSystem_::B, LinearSystem_::Init, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYS.

Referenced by approximate_system(), back_singleton(), CreateDefaultLinearSolution(), distribute_system(), eliminate_singletons(), flipsign(), LinearCreateNumericalSolution(), LinearSystemTrueDistance(), LinearSystemTrueDistancePrint(), MatGustafssonMod(), MatSymmetricPart(), sans_partition(), scale_system(), setup_-pc(), solvelinear(), specific_approximation_choices(), SysProComputeQuantity(), SysProRetrieveQuantity(), and unset_pc().

26.23.2.37 PetscErrorCode LinearSystemGetTmpVector (LinearSystem, Vec $*$)

Definition at line 265 of file linear.c.

References CHKERRQ(), ierr, LinearSystem_::Rhs, SYSPROCHECKVALIDLINSYS, and LinearSystem_::Tmp.

Referenced by LinearSystemTrueDistance(), and LinearSystemTrueDistancePrint().

26.23.2.38 PetscErrorCode LinearSystemInheritParts (LinearSystem *system*, Mat *A*, Mat *B*, Vec *Rhs*, Vec *Sol*, Vec *Init*)

Declare the matrices and vectors for a linear system. Unlike in LinearSystemSetParts(), here the parts are marked as not original, so they will not be deleted in DeleteLinearSystem().

Definition at line 162 of file linear.c.

References LinearSystem_::A, ALLPARTSNEW, LinearSystem_::B, CHKERRQ(), ierr, LinearSystem_::Init, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYS.

26.23.2.39 PetscErrorCode LinearSystemSetContext (LinearSystem, void $*$ *ctx*)

Definition at line 205 of file linear.c.

References LinearSystem_::ctx, and SYSPROCHECKVALIDLINSYS.

Referenced by eliminate_singletons().

26.23.2.40 PetscErrorCode LinearSystemSetKnownSolution (LinearSystem, PetscTruth)

Definition at line 225 of file linear.c.

References LinearSystem_::known_solution, and SYSPROCHECKVALIDLINSYS.

26.23.2.41 PetscErrorCode LinearSystemSetMetadata (LinearSystem, NMD_metadata)

Definition at line 245 of file linear.c.

References LinearSystem_::metadata, and SYSPROCHECKVALIDLINSYS.

Referenced by main().

26.23.2.42 PetscErrorCode LinearSystemSetParts (LinearSystem *system*, Mat *A*, Mat *B*, Vec *Rhs*, Vec *Sol*, Vec *Init*)

Declare the matrices and vectors for a linear system.

Arguments:

- `system`
- `A` : the matrix
- `B` : operator to construct the preconditioner from; if NULL, (or identical to A), A will be used
- rhs : right hand side
- sol : storage for the computed solution
- init : (optional) nontrivial starting vector for iterative solution

Definition at line 131 of file linear.c.

References LinearSystem_::A, LinearSystem_::B, CHKERRQ(), ierr, LinearSystem_-::Init, LinearSystem_::partsoriginal, LinearSystem_::Rhs, LinearSystem_::Sol, and SYSPROCHECKVALIDLINSYS.

Referenced by distribute_system(), eliminate_singletons(), flipsign(), main(), Mat-GustafssonMod(), MatSymmetricPart(), and setup_pc().

26.23.2.43 PetscErrorCode LinearSystemTrueDistance (LinearSystem, LinearSolution, PetscReal ∗)

Definition at line 652 of file linear.c.

References CHKERRQ(), ierr, LinearSolutionGetVector(), LinearSystemGetParts(), and LinearSystemGetTmpVector().

Referenced by LinearSystemTrueDistancePrint().

26.23.2.44 PetscErrorCode LinearSystemTrueDistancePrint (NumericalProblem, NumericalSolution, char ∗)

Definition at line 672 of file linear.c.

References CHKERRQ(), ierr, LinearSolutionGetVector(), LinearSystemGetKnown-Solution(), LinearSystemGetParts(), LinearSystemGetTmpVector(), LinearSys-temTrueDistance(), SYSPROCHECKVALIDLINSOL, and SYSPROCHECK-VALIDLINSYS.

26.23.2.45 PetscErrorCode make_diagnostics (char ∗, char ∗, Diagnostics ∗)

26.23.2.46 PetscErrorCode PreprocessedLinearSystemSolution (LinearSystem, LinearSolution ∗)

Definition at line 705 of file linear.c.

References CHKERRQ(), ierr, PreprocessedProblemSolving(), RegisterPreprocessor-Context(), and SYSPROCHECKVALIDLINSYS.

26.24 sysprosuit.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include "syspro.h"
#include "sysprolinear.h"
```

Functions

- PetscErrorCode onlyforsymmetricproblem (NumericalProblem, void ∗, SuitabilityValue ∗)

26.24.1 Function Documentation

26.24.1.1 PetscErrorCode onlyforsymmetricproblem (NumericalProblem *problem*, void ∗ *ctx*, SuitabilityValue ∗ *v*)

26.24.2 Suitability functions for the linear problem

Definition at line 18 of file suit.c.

References CHKERRQ(), and ierr.

Referenced by setup_ksp_choices().

26.25 sysprotransform.h File Reference

```
#include "syspro.h"
#include "petsc.h"
```

Functions

- PetscErrorCode NewTransform (char ∗name, SalsaTransform ∗)
- PetscErrorCode DeregisterTransform (SalsaTransform)
- PetscErrorCode TransformGetName (SalsaTransform, char ∗∗)
- PetscErrorCode TransformGetByName (char ∗, SalsaTransform ∗)
- PetscErrorCode NewTransformObject (char ∗, char ∗, SalsaTransformObject ∗)
- PetscErrorCode FreeTransformObject (SalsaTransformObject tf)
- PetscErrorCode TransformObjectGetByName (char ∗, char ∗, SalsaTransformObject ∗)
- PetscErrorCode TransformGetObjects (SalsaTransform, int ∗, SalsaTransformObject ∗∗)
- PetscErrorCode TransformObjectGetName (SalsaTransformObject, char ∗∗)
- PetscErrorCode TransformObjectGetTransformName (SalsaTransformObject, char ∗∗)
- PetscErrorCode TransformObjectsGetNames (SalsaTransform, char ∗∗∗)
- PetscErrorCode TransformObjectSetExplanation (SalsaTransformObject, char ∗)

- PetscErrorCode TransformObjectSetSuitabilityFunction (SalsaTransformObject, void ∗, PetscErrorCode(∗)(NumericalProblem, void ∗, SuitabilityValue ∗))
- PetscErrorCode TransformObjectGetSuitabilityFunction (SalsaTransformObject, void ∗∗, PetscErrorCode(∗∗)(NumericalProblem, void ∗, SuitabilityValue ∗))
- PetscErrorCode TransformObjectAddOptionExplanation (SalsaTransformObject, int, char ∗)
- PetscErrorCode TransformItemGetFirstOption (char ∗, char ∗, int ∗, PetscTruth ∗)
- PetscErrorCode TransformItemGetNextOption (char ∗, char ∗, int ∗, PetscTruth ∗)
- PetscErrorCode TransformReportTeXTable (SalsaTransform, FILE ∗)
- PetscErrorCode TransformReportEnabled (SalsaTransform, char ∗∗)
- PetscErrorCode TransformItemDescribeShort (SalsaTransform, char ∗, int, char ∗∗)
- PetscErrorCode TransformItemDescribeLong (SalsaTransform, char ∗, int, char ∗∗)
- PetscErrorCode TransformObjectMark (SalsaTransformObject tf)
- PetscErrorCode TransformObjectUnmark (SalsaTransformObject tf)
- PetscErrorCode TransformObjectsMarkAll (SalsaTransform tf)
- PetscErrorCode TransformObjectsUnmarkAll (SalsaTransform tf)
- PetscErrorCode TransformObjectGetMark (SalsaTransformObject, int ∗)
- PetscErrorCode TransformItemOptionMark (SalsaTransform, char ∗, int)
- PetscErrorCode TransformGetNUnmarked (SalsaTransform, int ∗)
- PetscErrorCode TransformObjectsUseOnly (SalsaTransform, char ∗list)
- PetscErrorCode TransformItemOptionsUseOnly (SalsaTransformObject, char ∗)
- PetscErrorCode TransformGetNItems (SalsaTransform, int ∗n)
- PetscErrorCode TransformGetNextUnmarkedItem (SalsaTransform, char ∗, SalsaTransformObject ∗, PetscTruth ∗)
- PetscErrorCode PreprocessorSaveAprioriSelection (SystemPreprocessor)
- PetscErrorCode PreprocessorApplyAprioriSelection (SystemPreprocessor)
- PetscErrorCode SysProDefineCharAnnotation (char ∗, char ∗)
- PetscErrorCode TransformCharAnnotationGetIndex (SalsaTransform, char ∗, int ∗, PetscTruth ∗)
- PetscErrorCode TransformObjectCharAnnotate (SalsaTransformObject, char ∗, char ∗)
- PetscErrorCode TransformObjectIntAnnotate (SalsaTransformObject tf, char ∗, int)
- PetscErrorCode TransformObjectGetIntAnnotation (SalsaTransformObject, char ∗an, int ∗v, PetscTruth ∗f)
- PetscErrorCode SysProDefineIntAnnotation (char ∗, char ∗)
- PetscErrorCode TransformIntAnnotationGetIndex (SalsaTransform, char ∗, int ∗, PetscTruth ∗)

- PetscErrorCode TransformItemIntAnnotate (SalsaTransform, int, int)
- PetscErrorCode TransformItemGetIntAnnotation (SalsaTransform, int idx, char ∗an, int ∗v, PetscTruth ∗f)
- PetscErrorCode TransformObjectDefineOption (SalsaTransformObject, char ∗)
- PetscErrorCode TransformObjectAddOption (SalsaTransformObject, int)
- PetscErrorCode TransformItemCharAnnotationGetIndex (SalsaTransform, char ∗, int ∗)
- PetscErrorCode TransformItemGetCharAnnotation (SalsaTransform, int idx, char ∗an, char ∗∗v, PetscTruth ∗)
- PetscErrorCode TransformCurrentItemDefineOption (SalsaTransform, char ∗, char ∗)
- PetscErrorCode TransformItemDefineOption (SalsaTransform, int, char ∗, char ∗)
- PetscErrorCode TransformItemGetNOptions (SalsaTransform, int it, int ∗nopt)
- PetscErrorCode TransformItemGetOptionI (SalsaTransform, int it, int iopt, int ∗v)
- PetscErrorCode TransformSetUserChoices (SalsaTransform, PetscTruth)
- PetscErrorCode TransformGetUserChoices (SalsaTransform, PetscTruth ∗)

26.25.1 Function Documentation

26.25.1.1 PetscErrorCode DeregisterTransform (SalsaTransform)

Definition at line 49 of file transform.c.

References SalsaTransform_::annotations_c, SalsaTransform_::annotations_i, SalsaTransform_::aprioriselection, CHKERRQ(), FreeTransformObject(), ierr, SalsaTransform_::n_objects, and SalsaTransform_::transformobjects.

Referenced by SysProFinalize().

26.25.1.2 PetscErrorCode FreeTransformObject (SalsaTransformObject *tf*)

Definition at line 109 of file transform.c.

References SalsaTransformObject_::annotate_c, SalsaTransformObject_::annotate_-i, CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_-::name, SalsaTransformObject_::optionexplanation, SalsaTransformObject_::options, and SalsaTransformObject_::options_marked.

Referenced by DeregisterTransform().

26.25.1.3 PetscErrorCode NewTransform (char ∗ *name*, SalsaTransform ∗ *tf*)

Define a new class of preprocessors, for instance scaling or permutation.

Definition at line 31 of file transform.c.

References SalsaTransform_::alloc_objects, SalsaTransform_::aprioriselection, CHK-ERRQ(), ierr, SalsaTransform_::n_objects, SalsaTransform_::name, TFINC, and SalsaTransform_::transformobjects.

Referenced by DeclarePreprocessor().

26.25.1.4 PetscErrorCode NewTransformObject (char ∗ *transform*, char ∗ *name*, SalsaTransformObject ∗ *to*)

Create a transform object specified by `name` for the preprocessor class `transform`. The `to` parameter can be NULL if no further specifications of the object are needed, in which case this only registers the name.

Definition at line 92 of file transform.c.

References SalsaTransform_::alloc_objects, CHKERRQ(), ierr, SalsaTransform_::n_-objects, SalsaTransformObject_::name, SalsaTransformObject_::transform, Transfor-mGetByName(), and SalsaTransform_::transformobjects.

Referenced by declareadders(), setup_approximation_choices(), setup_distribution_-choices(), setup_flipsign_choices(), setup_ksp_choices(), setup_pc_choices(), setup_-scaling_choices(), and setup_singleton_choices().

26.25.1.5 PetscErrorCode PreprocessorApplyAprioriSelection (SystemPreprocessor)

Definition at line 426 of file transform.c.

References SalsaTransform_::aprioriselection, SalsaTransformObject_::marked, SalsaTransform_::n_objects, SystemPreprocessor_::transform, and SalsaTransform_-::transformobjects.

Referenced by PreprocessorSpecificSetup().

26.25.1.6 PetscErrorCode PreprocessorSaveAprioriSelection (SystemPreprocessor)

Definition at line 414 of file transform.c.

References SalsaTransform_::aprioriselection, SalsaTransformObject_::marked, SalsaTransform_::n_objects, SystemPreprocessor_::transform, and SalsaTransform_-::transformobjects.

Referenced by PreprocessorsOptionsHandling().

26.25.1.7 PetscErrorCode SysProDefineCharAnnotation (char $*$ *transform*, char $*$ *ann*)

Define a character string annotation for a transform. The index of this annotation can be retrieved with TransformCharAnnotationGetIndex(). The actual annotation can be found with TransformItemGetCharAnnotation().

Definition at line 243 of file transform.c.

References SalsaTransform_::annotations_c, CHKERRQ(), ierr, SalsaTransform_::n_-annotate_c, TFINC, and TransformGetByName().

Referenced by setup_ksp_choices().

26.25.1.8 PetscErrorCode SysProDefineIntAnnotation (char $*$ *transform*, char $*$ *ann*)

Define a integer string annotation for a transform. The index of this annotation can be retrieved with TransformIntAnnotationGetIndex(). The actual annotation can be found with TransformItemGetIntAnnotation().

Definition at line 297 of file transform.c.

References SalsaTransform_::annotations_i, CHKERRQ(), ierr, SalsaTransform_::n_-annotate_i, TFINC, and TransformGetByName().

Referenced by setup_distribution_choices(), setup_ksp_choices(), setup_pc_choices(), and setup_scaling_choices().

26.25.1.9 PetscErrorCode TransformCharAnnotationGetIndex (SalsaTransform, char ∗, int ∗, PetscTruth ∗)

Definition at line 277 of file transform.c.

References SalsaTransform_::annotations_c, CHKERRQ(), ierr, and SalsaTransform_-::n_annotate_c.

Referenced by TransformObjectCharAnnotate().

26.25.1.10 PetscErrorCode TransformCurrentItemDefineOption (SalsaTransform, char ∗, char ∗)

26.25.1.11 PetscErrorCode TransformGetByName (char ∗, SalsaTransform ∗)

Definition at line 611 of file preprocess.c.

References CHKERRQ(), ierr, SystemPreprocessorGetByName(), and SystemPreprocessor_::transform.

Referenced by NewTransformObject(), PreprocessorsOptionsHandling(), PreprocessorSpecificSetup(), ReportEnabledPreprocessors(), setup_approximation_choices(), setup_distribution_choices(), setup_flipsign_choices(), setup_ksp_choices(), setup_-pc_choices(), setup_scaling_choices(), setup_singleton_choices(), SysProDefineCharAnnotation(), SysProDefineIntAnnotation(), and TransformObjectGetByName().

26.25.1.12 PetscErrorCode TransformGetName (SalsaTransform, char ∗∗)

Definition at line 66 of file transform.c.

References SalsaTransform_::name.

26.25.1.13 PetscErrorCode TransformGetNextUnmarkedItem (SalsaTransform *tf*, char ∗ *old*, SalsaTransformObject ∗ *snew*, PetscTruth ∗ *f*)

Find the next unmarked value; if `old` is NULL, the first first unmarked value is given, otherwise the first one after a match with `old`.

Definition at line 477 of file transform.c.

References SalsaTransformObject_::marked, SalsaTransform_::n_objects, SalsaTransformObject_::name, SalsaTransform_::transformobjects, and TRUTH.

Referenced by ChooseFirstTransform(), PreprocessedSolution(), and PreprocessorsOptionsHandling().

26.25.1.14 PetscErrorCode TransformGetNItems (SalsaTransform, int ∗ *n*)

26.25.1.15 PetscErrorCode TransformGetNUnmarked (SalsaTransform, int ∗)

Definition at line 402 of file transform.c.

References SalsaTransformObject_::marked, SalsaTransform_::n_objects, and SalsaTransform_::transformobjects.

Referenced by PreprocessorsOptionsHandling().

26.25.1.16 PetscErrorCode TransformGetObjects (SalsaTransform, int ∗, SalsaTransformObject ∗∗)

Definition at line 76 of file transform.c.

References SalsaTransform_::n_objects, and SalsaTransform_::transformobjects.

Referenced by disable_ksps(), specific_distribution_choices(), and specific_scaling_-choices().

26.25.1.17 PetscErrorCode TransformGetUserChoices (SalsaTransform, PetscTruth ∗)

Definition at line 794 of file transform.c.

References SalsaTransform_::userchoices.

Referenced by PreprocessedSolution().

26.25.1.18 PetscErrorCode TransformIntAnnotationGetIndex (SalsaTransform, char ∗, int ∗, PetscTruth ∗)

Definition at line 329 of file transform.c.

References SalsaTransform_::annotations_i, CHKERRQ(), ierr, and SalsaTransform_-::n_annotate_i.

Referenced by TransformObjectIntAnnotate().

26.25.1.19 PetscErrorCode TransformItemCharAnnotationGetIndex (SalsaTransform, char ∗, int ∗)

26.25.1.20 PetscErrorCode TransformItemDefineOption (SalsaTransform, int, char ∗, char ∗)

26.25.1.21 PetscErrorCode TransformItemDescribeLong (SalsaTransform, char ∗, int, char ∗∗)

Definition at line 755 of file transform.c.

References CHKERRQ(), SalsaTransformObject_::explanation, ierr, SalsaTransform_::name, SalsaTransformObject_::options, and TransformObject-GetByName().

26.25.1.22 PetscErrorCode TransformItemDescribeShort (SalsaTransform, char ∗, int, char ∗∗)

Definition at line 737 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_::name, SalsaTransform_::name, and TransformObject-GetByName().

26.25.1.23 PetscErrorCode TransformItemGetCharAnnotation (SalsaTransform, int *idx*, char $*$ *an*, char $**$ *v*, PetscTruth $*$)

26.25.1.24 PetscErrorCode TransformItemGetFirstOption (char $*$, char $*$, int $*$, PetscTruth $*$)

Definition at line 572 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_::options, SalsaTransformObject_::options_marked, and TransformObjectGetByName().

Referenced by PreprocessedSolution().

26.25.1.25 PetscErrorCode TransformItemGetIntAnnotation (SalsaTransform, int *idx*, char $*$ *an*, int $*$ *v*, PetscTruth $*$ *f*)

26.25.1.26 PetscErrorCode TransformItemGetNextOption (char $*$, char $*$, int $*$, PetscTruth $*$)

Definition at line 595 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_::options, SalsaTransformObject_::options_marked, and TransformObjectGetByName().

Referenced by PreprocessedSolution().

26.25.1.27 PetscErrorCode TransformItemGetNOptions (SalsaTransform, int *it*, int $*$ *nopt*)

26.25.1.28 PetscErrorCode TransformItemGetOptionI (SalsaTransform, int *it*, int *iopt*, int $*$ *v*)

26.25.1.29 PetscErrorCode TransformItemIntAnnotate (SalsaTransform, int, int)

26.25.1.30 PetscErrorCode TransformItemOptionMark (SalsaTransform, char *, int)

Definition at line 619 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::marked, SalsaTransformObject_::n_options, SalsaTransform_::name, SalsaTransformObject_-::options, SalsaTransformObject_::options_marked, and TransformObjectGetBy-Name().

26.25.1.31 PetscErrorCode TransformItemOptionsUseOnly (SalsaTransformObject, char *)

Definition at line 555 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::n_options, and TransformOb-jectAddOption().

Referenced by PreprocessorsOptionsHandling().

26.25.1.32 PetscErrorCode TransformObjectAddOption (SalsaTransformObject, int)

Definition at line 517 of file transform.c.

References SalsaTransformObject_::alloc_options, CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_::name, SalsaTransformObject_::optionexplanation, SalsaTransformObject_::options, SalsaTransformObject_::options_marked, and TFINC.

Referenced by declareadders(), setup_ksp_choices(), setup_pc_choices(), and Trans-formItemOptionsUseOnly().

26.25.1.33 PetscErrorCode TransformObjectAddOptionExplanation (SalsaTransformObject, int, char $*$)

Definition at line 539 of file transform.c.

References SalsaTransformObject_::n_options, SalsaTransformObject_-::optionexplanation, and SalsaTransformObject_::options.

Referenced by setup_pc_choices().

26.25.1.34 PetscErrorCode TransformObjectCharAnnotate (SalsaTransformObject, char $*$, char $*$)

Definition at line 634 of file transform.c.

References SalsaTransformObject_::alloc_annotate_c, SalsaTransformObject_-::annotate_c, CHKERRQ(), ierr, TFINC, SalsaTransformObject_::transform, and TransformCharAnnotationGetIndex().

26.25.1.35 PetscErrorCode TransformObjectDefineOption (SalsaTransformObject, char $*$)

Definition at line 508 of file transform.c.

References SalsaTransformObject_::option.

Referenced by declareadders(), setup_ksp_choices(), and setup_pc_choices().

26.25.1.36 PetscErrorCode TransformObjectGetByName (char $*$, char $*$, SalsaTransformObject $*$)

Definition at line 202 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, SalsaTransformObject_-::name, TransformGetByName(), and SalsaTransform_::transformobjects.

Referenced by disable_ksps(), disable_pcs(), is_gmres_method(), pcoptionshandling(), setup_ksp(), specific_approximation_choices(), specific_flipsign_choices(), specific_-singleton_choices(), TransformItemDescribeLong(), TransformItemDescribeShort(), TransformItemGetFirstOption(), TransformItemGetNextOption(), TransformItemOp-tionMark(), TransformObjectsUseOnly(), and unset_ksps().

26.25.1.37 PetscErrorCode TransformObjectGetIntAnnotation (SalsaTransformObject, char $*$ *an*, int $*$ *v*, PetscTruth $*$ *f*)

Definition at line 676 of file transform.c.

References SalsaTransformObject_::annotate_i, SalsaTransform_::annotations_i, SalsaTransform_::n_annotate_i, SalsaTransformObject_::transform, and TRUTH.

Referenced by disable_ksps(), is_gmres_method(), pcoptionshandling(), set_ksp_-options(), specific_distribution_choices(), and specific_scaling_choices().

26.25.1.38 PetscErrorCode TransformObjectGetMark (SalsaTransformObject, int $*$)

Definition at line 393 of file transform.c.

References SalsaTransformObject_::marked.

Referenced by TransformReportEnabled().

26.25.1.39 PetscErrorCode TransformObjectGetName (SalsaTransformObject, char $**$)

Definition at line 130 of file transform.c.

References SalsaTransformObject_::name.

Referenced by ChooseFirstTransform(), disable_ksps(), PreprocessedSolution(), PreprocessorsOptionsHandling(), and set_ksp_options().

26.25.1.40 PetscErrorCode TransformObjectGetSuitabilityFunction (SalsaTransformObject *tf*, void $**$ *sctx*, PetscErrorCode($**$)(NumericalProblem, void $*$, SuitabilityValue $*$) *f*)

Retrieve the suitability function and context; see Suitability functions. Both arguments can be null.

Definition at line 182 of file transform.c.

References SalsaTransformObject_::suitabilityctx, and SalsaTransformObject_-::suitabilityfunction.

Referenced by PreprocessorSpecificSetup(), and unset_ksps().

26.25.1.41 PetscErrorCode TransformObjectGetTransformName (SalsaTransformObject, char $**$)

Definition at line 192 of file transform.c.

References SalsaTransform_::name, and SalsaTransformObject_::transform.

26.25.1.42 PetscErrorCode TransformObjectIntAnnotate (SalsaTransformObject *tf*, char $*$, int)

Definition at line 655 of file transform.c.

References SalsaTransformObject_::alloc_annotate_i, SalsaTransformObject_-::annotate_i, CHKERRQ(), ierr, TFINC, SalsaTransformObject_::transform, and TransformIntAnnotationGetIndex().

Referenced by setup_distribution_choices(), setup_ksp_choices(), setup_pc_choices(), and setup_scaling_choices().

26.25.1.43 PetscErrorCode TransformObjectMark (SalsaTransformObject *tf*)

Definition at line 345 of file transform.c.

References SalsaTransformObject_::marked, SalsaTransformObject_::n_options, and SalsaTransformObject_::options_marked.

Referenced by disable_ksps(), disable_pcs(), pcoptionshandling(), Preprocessor-SpecificSetup(), specific_approximation_choices(), specific_distribution_choices(), specific_flipsign_choices(), specific_scaling_choices(), specific_singleton_choices(), TransformObjectsMarkAll(), and TransformObjectsUseOnly().

26.25.1.44 PetscErrorCode TransformObjectSetExplanation (SalsaTransformObject, char $*$)

Definition at line 139 of file transform.c.

References SalsaTransformObject_::explanation.

Referenced by setup_approximation_choices(), setup_distribution_choices(), setup_-flipsign_choices(), setup_ksp_choices(), setup_pc_choices(), setup_scaling_choices(), and setup_singleton_choices().

26.25.1.45 PetscErrorCode TransformObjectSetSuitabilityFunction (SalsaTransformObject *tf*, void ∗ *sctx*, PetscErrorCode(∗)(NumericalProblem, void ∗, SuitabilityValue ∗) *f*)

Set the suitability function; see Suitability functions

Definition at line 168 of file transform.c.

References SalsaTransformObject_::suitabilityctx, and SalsaTransformObject_-::suitabilityfunction.

Referenced by setup_ksp_choices().

26.25.1.46 PetscErrorCode TransformObjectsGetNames (SalsaTransform *tf*, char ∗∗∗ *names*)

Get the names of all declared transformobjects. An array is allocated for the names, which needs to be PetscFree()'d.

Definition at line 223 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, SalsaTransformObject_-::name, and SalsaTransform_::transformobjects.

Referenced by ContinueRetrievingAllPreprocessors(), and TransformReportEn-abled().

26.25.1.47 PetscErrorCode TransformObjectsMarkAll (SalsaTransform *tf*)

Definition at line 369 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, TransformObjectMark(), and SalsaTransform_::transformobjects.

Referenced by TransformObjectsUseOnly().

26.25.1.48 PetscErrorCode TransformObjectsUnmarkAll (SalsaTransform *tf*)

Definition at line 381 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, SalsaTransform_-::transformobjects, and TransformObjectUnmark().

Referenced by disable_ksps(), and TransformObjectsUseOnly().

26.25.1.49 PetscErrorCode TransformObjectsUseOnly (SalsaTransform *tf*, char ∗ *list*)

Mark a list of names as to be used.

Cases:

- `"name1,name2,name3"` : all other names are marked as not to be used
- `"not,name1,name2"` : all names will be used, except for the ones listed

Definition at line 445 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::name, TransformObjectGetByName(), TransformObjectMark(), TransformObjectsMarkAll(), TransformObjectsUnmarkAll(), and TransformObjectUnmark().

Referenced by PreprocessorsOptionsHandling().

26.25.1.50 PetscErrorCode TransformObjectUnmark (SalsaTransformObject *tf*)

Definition at line 357 of file transform.c.

References SalsaTransformObject_::marked, SalsaTransformObject_::n_options, and SalsaTransformObject_::options_marked.

Referenced by TransformObjectsUnmarkAll(), and TransformObjectsUseOnly().

26.25.1.51 PetscErrorCode TransformReportEnabled (SalsaTransform, char ∗∗)

Definition at line 715 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, TransformObjectGetMark(), SalsaTransform_::transformobjects, and TransformObjectsGetNames().

Referenced by ReportEnabledPreprocessors().

26.25.1.52 PetscErrorCode TransformReportTeXTable (SalsaTransform, FILE ∗)

Definition at line 693 of file transform.c.

References SalsaTransformObject_::explanation, SalsaTransform_::n_-objects, SalsaTransformObject_::n_options, SalsaTransformObject_::name, SalsaTransformObject_::options, and SalsaTransform_::transformobjects.

26.25.1.53 PetscErrorCode TransformSetUserChoices (SalsaTransform, PetscTruth)

Definition at line 785 of file transform.c.

References SalsaTransform_::userchoices.

Referenced by PreprocessorsOptionsHandling().

26.26 testmat.c File Reference

Functions

- CHKERRQ (ierr)
- for (i=0;i< n;i++)

Variables

- ierr = MatCreateSeqAIJ(MPI_COMM_SELF,n,n,3,0,&A)

26.26.1 Function Documentation

26.26.1.1 CHKERRQ (ierr)

Referenced by adder(), approximate_system(), back_flipsign(), back_singleton(), ChooseFirstTransform(), ContinueRetrievingAllPreprocessors(), ContinueRetrievingCurrentPreprocessors(), copy(), create_solver(), CreateDefaultLinearSolution(), CreateGlobalInfo(), CreateLinearSolution(), CreateLinearSystem(), declareadders(), DeclareApproximationPreprocessor(), DeclareDistributionPreprocessor(), DeclareFlipsignPreprocessor(), DeclareKSPPreprocessor(), DeclarePCPreprocessor(), DeclarePreprocessor(), DeclarePreprocessorIntelligentChoice(), DeclarePreprocessorRequiredCategories(), DeclareScalingPreprocessor(), DeclareSingletonPreprocessor(), DeleteLinearSystem(), delintctx(), delprob(), delsol(), DeregisterTransform(), destroy_solver(), destroysolution(), disable_ksps(), disable_pcs(), distribute_system(), eliminate_singletons(), flipsign(), for(), FreeTransformObject(), get_pc_stats_-function(), GetFirstPreprocessor(), is_gmres_method(), LinearCopyNumericalSolution(), LinearCreateNumericalSolution(), LinearDeleteNumericalSolution(), LinearSolutionCopy(), LinearSolutionCopyStats(), LinearSolutionCreateStatistics(), LinearSolutionDelete(), LinearSystemCopy(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetTmpVector(), LinearSystemInheritParts(), LinearSystemSetParts(), LinearSystemTrueDistance(), LinearSystemTrueDistancePrint(), main(), makeintctx(), makesol(), MatGustafssonMod(), MatSymmetricPart(), MonitorAdjustMaxit(), NewTransform(), NewTransformObject(), onlyforsymmetricproblem(), pc_short_string(), pc_string(), pcoptionshandling(), PreprocessedLinearSystemSolution(), PreprocessedProblemSolving(), PreprocessedSolution(), PreprocessorGetContext(), PreprocessorGetIndex(), PreprocessorGetPreservedCategories(), PreprocessorSetPreservedCategories(), PreprocessorsOptionsHandling(), PreprocessorSpecificSetup(), RegisterPreprocessorContext(), ReportEnabledPreprocessors(), ReportSysProCallStackState(), RetrieveAllPreprocessorValues(), sans_partition(), scale_system(), ScreenOutputTab(), ScreenOutputTabLine(), set_blocked_sub_pc(), set_intelligent_scaling(), set_ksp_options(), set_preconditioner_base_matrix(), SetPetscOptionsForPC(), setup_approximation_-choices(), setup_distribution_choices(), setup_flipsign_choices(), setup_ksp(), setup_ksp_choices(), setup_pc(), setup_pc_choices(), setup_scaling_choices(), setup_singleton_choices(), singleton_specific_unset(), solvebycopy(), solvelinear(), specific_approximation_choices(), specific_distribution_choices(), specific_flipsign_-choices(), specific_scaling_choices(), specific_singleton_choices(), StartRetrievingAllPreprocessors(), StartRetrievingCurrentPreprocessors(), SuccessorPreprocessor(), SysProComputeQuantity(), SysProDefineCharAnnotation(), SysProDefineIntAnnotation(), SysProFinalize(), SysProFreeQuantities(), SysProInitialize(), SysProLinearInstallCustomKSPMonitor(), SysProPreprocessorEndFunction(), SysproPreprocessorStartFunction(), SysProProblemCloneContext(), SysProProblemDeleteContext(), SysProRemoveQuantity(), SysProRetrieveQuantity(), SysProTraceMessage(), SystemPreprocessorGetByName(), TabReportActivePreprocessors(), TabReportAllPreprocessors(), TabReportPreprocessors(), TransformCharAnnotationGetIndex(), TransformGetByName(), TransformIntAnnotationGetIndex(), TransformItemDescribeLong(), TransformItemDescribeShort(), TransformItemGetFirstOption(), TransformItemGetNextOption(), TransformItemOptionMark(), TransformItemOptionsUseOnly(), TransformObjectAddOption(), TransformObjectCharAnnotate(), TransformObjectGetByName(), TransformObjectIntAnnotate(), TransformObjectsGetNames(), TransformObjectsMarkAll(), TransformObjectsUnmarkAll(), TransformObjectsUseOnly(), TransformReportEnabled(), unapproximate_system(), undistribute_system(), unscale_system(), unset_ksp(), unset_ksps(), and unset_pc().

26.26.1.2 for ()

Definition at line 6 of file testmat.c.

References CHKERRQ(), and ierr.

26.26.2 Variable Documentation

26.26.2.1 ierr = MatCreateSeqAIJ(MPI_COMM_SELF,n,n,3,0,&A)

Definition at line 4 of file testmat.c.

Referenced by adder(), approximate_system(), back_flipsign(), back_singleton(), ChooseFirstTransform(), ContinueRetrievingAllPreprocessors(), ContinueRetrievingCurrentPreprocessors(), copy(), create_solver(), CreateDefaultLinearSolution(), CreateGlobalInfo(), CreateLinearSolution(), CreateLinearSystem(), declareadders(), DeclareApproximationPreprocessor(), DeclareDistributionPreprocessor(), DeclareFlipsignPreprocessor(), DeclareKSPPreprocessor(), DeclarePCPreprocessor(), DeclarePreprocessor(), DeclarePreprocessorIntelligentChoice(), DeclarePreprocessorRequiredCategories(), DeclareScalingPreprocessor(), DeclareSingletonPreprocessor(), DeleteLinearSystem(), delintctx(), delprob(), delsol(), DeregisterTransform(), destroy_solver(), destroysolution(), disable_ksps(), disable_pcs(), distribute_system(), eliminate_singletons(), flipsign(), for(), FreeTransformObject(), GetFirstPreprocessor(), is_gmres_method(), LinearCopyNumericalSolution(), LinearCreateNumericalSolution(), LinearDeleteNumericalSolution(), LinearSolutionCopy(), LinearSolutionCopyStats(), LinearSolutionCreateStatistics(), LinearSolutionDelete(), LinearSystemCopy(), LinearSystemDuplicate(), LinearSystemDuplicatePointers(), LinearSystemGetTmpVector(), LinearSystemInheritParts(), LinearSystemSetParts(), LinearSystemTrueDistance(), LinearSystemTrueDistancePrint(), main(), makeintctx(), makesol(), MatGustafssonMod(), MatSymmetricPart(), MonitorAdjustMaxit(), NewTransform(), NewTransformObject(), onlyforsymmetricproblem(), pc_short_string(), pc_string(), pcoptionshandling(), PreprocessedLinearSystemSolution(), PreprocessedProblemSolving(), PreprocessedSolution(), PreprocessorGetContext(), PreprocessorGetIndex(), PreprocessorGetPreservedCategories(), PreprocessorSetPreservedCategories(), PreprocessorsOptionsHandling(), PreprocessorSpecificSetup(), RegisterPreprocessorContext(), ReportEnabledPreprocessors(), ReportSysProCallStackState(), RetrieveAllPreprocessorValues(), sans_partition(), scale_system(), ScreenOutputTab(), ScreenOutputTabLine(), set_blocked_sub_pc(), set_intelligent_scaling(), set_ksp_options(), setup_approximation_choices(), setup_distribution_choices(), setup_flipsign_choices(), setup_ksp(), setup_ksp_choices(), setup_pc(), setup_pc_choices(), setup_scaling_choices(), setup_singleton_choices(), singleton_specific_unset(), solvebycopy(), solvelinear(), specific_approximation_choices(), specific_distribution_choices(), specific_flipsign_choices(), specific_-

scaling_choices(), specific_singleton_choices(), StartRetrievingAllPreprocessors(), StartRetrievingCurrentPreprocessors(), SuccessorPreprocessor(), SysProComputeQuantity(), SysProDefineCharAnnotation(), SysProDefineIntAnnotation(), SysProFinalize(), SysProFreeQuantities(), SysProInitialize(), SysProLinearInstallCustomKSPMonitor(), SysProPreprocessorEndFunction(), SysproPreprocessorStartFunction(), SysProProblemCloneContext(), SysProProblemDeleteContext(), SysProRemoveQuantity(), SysProRetrieveQuantity(), SysProTraceMessage(), SystemPreprocessorGetByName(), TabReportActivePreprocessors(), TabReportAllPreprocessors(), TabReportPreprocessors(), TransformCharAnnotationGetIndex(), TransformGetByName(), TransformIntAnnotationGetIndex(), TransformItemDescribeLong(), TransformItemDescribeShort(), TransformItemGetFirstOption(), TransformItemGetNextOption(), TransformItemOptionMark(), TransformItemOptionsUseOnly(), TransformObjectAddOption(), TransformObjectCharAnnotate(), TransformObjectGetByName(), TransformObjectIntAnnotate(), TransformObjectsGetNames(), TransformObjectsMarkAll(), TransformObjectsUnmarkAll(), TransformObjectsUseOnly(), TransformReportEnabled(), unapproximate_system(), undistribute_system(), unscale_system(), unset_ksp(), unset_ksps(), and unset_pc().

26.27 testmat16.c File Reference

Functions

- CHKERRQ (ierr)
- for (i=0;i< n;i++)

Variables

- ierr = MatCreateSeqAIJ(MPI_COMM_SELF,n,n,3,0,&A)

26.27.1 Function Documentation

26.27.1.1 CHKERRQ (ierr)

26.27.1.2 for ()

Definition at line 6 of file testmat16.c.

References CHKERRQ(), and ierr.

26.27.2 Variable Documentation

26.27.2.1 ierr = MatCreateSeqAIJ(MPI_COMM_SELF,n,n,3,0,&A)

Definition at line 4 of file testmat16.c.

26.28 tracing.c File Reference

```
#include <stdlib.h>
#include <string.h>
#include <stdarg.h>
#include "syspro.h"
```

Functions

- PetscErrorCode SysProDefaultTrace (void ∗ctx, char ∗fmt, va_list argp)
- PetscErrorCode SysProDeclareTraceFunction (PetscErrorCode(∗fn)(void ∗, char ∗, va_list))
- PetscErrorCode SysProDeclareTraceContext (void ∗ctx)
- PetscErrorCode SysProTraceMessage (char ∗fmt,...)
- PetscErrorCode SysProHasTrace (PetscTruth ∗flg)

Variables

- static PetscErrorCode(∗ sysprotrace)(void ∗, char ∗fmt, va_list) = NULL
- static size_t sysprotracectx = (size_t)NULL

26.28.1 Function Documentation

26.28.1.1 PetscErrorCode SysProDeclareTraceContext (void ∗ *ctx*)

Definition at line 77 of file tracing.c.

References sysprotracectx.

26.28.1.2 PetscErrorCode SysProDeclareTraceFunction (PetscErrorCode(∗)(void ∗, char ∗, va_list) *fn*)

Specify a trace function.

The trace function has a prototype

```
  PetscErrorCode tracefunction(void*,char*,va_list)
```

which means that it has an arbitrary number of arguments, much like `printf`. The first argument is a context, which can be set by SysProDeclareTraceContext().

Here is an example of how you would write a trace function:

```
#include <stdarg.h>
PetscErrorCode tracefunction(void *ctx,char *fmt,va_list argp)
{
  char *prefix = (char*)ctx;
  PetscFunctionBegin;
  printf("%s ",prefix);
  vprintf(fmt, argp);
  PetscFunctionReturn(0);
}
```

Consult `string.h` (probably in `/usr/include`) to see which "v" versions of `printf` are available.

There is a default trace function SysProDefaultTrace().

You can undeclare a trace function by passing NULL.

See also SysProTraceMessage().

Definition at line 64 of file tracing.c.

References sysprotrace.

Referenced by main().

26.28.1.3 PetscErrorCode SysProDefaultTrace (void ∗ *ctx*, char ∗ *fmt*, va_list *argp*)

Definition at line 22 of file tracing.c.

Referenced by main().

26.28.1.4 PetscErrorCode SysProHasTrace (PetscTruth ∗ *flg*)

Test whether a trace function has been declared; see SysProDeclareTraceFunction(). Normally you would use SysProTraceMessage() which performs this test internally, but this function can be useful if a large amount of processing has to be performed to construct the trace message to begin with.

Definition at line 109 of file tracing.c.

References sysprotrace.

Referenced by ReportEnabledPreprocessors(), ReportSysProCallStackState(), ScreenOutputTab(), and ScreenOutputTabLine().

26.28.1.5 PetscErrorCode SysProTraceMessage (char ∗ *fmt*, ...)

This function prints a trace message if a trace function has been declared; see SysProDeclareTraceFunction().

Definition at line 89 of file tracing.c.

References CHKERRQ(), ierr, sysprotrace, and sysprotracectx.

Referenced by adder(), ChooseFirstTransform(), PreprocessedSolution(), ReportEnabledPreprocessors(), ReportSysProCallStackState(), ScreenOutputTab(), ScreenOutputTabLine(), and solvebycopy().

26.28.2 Variable Documentation

26.28.2.1 PetscErrorCode(∗ sysprotrace)(void ∗, char ∗fmt, va_list) = NULL `[static]`

Referenced by SysProDeclareTraceFunction(), SysProHasTrace(), and SysProTraceMessage().

26.28.2.2 size_t sysprotracectx = (size_t)NULL `[static]`

Definition at line 18 of file tracing.c.

Referenced by SysProDeclareTraceContext(), and SysProTraceMessage().

26.29 transform.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "petsc.h"
#include "sysprotransform.h"
#include "syspro_impl.h"
```

Defines

- #define TFINC 20
- #define STRDUP(a) ((a) ? strdup(a) : NULL)

Functions

- PetscErrorCode NewTransform (char ∗name, SalsaTransform ∗tf)
- PetscErrorCode DeregisterTransform (SalsaTransform tf)
- PetscErrorCode TransformGetName (SalsaTransform tf, char ∗∗name)
- PetscErrorCode TransformGetObjects (SalsaTransform tf, int ∗n, SalsaTransformObject ∗∗objs)
- PetscErrorCode NewTransformObject (char ∗transform, char ∗name, SalsaTransformObject ∗to)
- PetscErrorCode FreeTransformObject (SalsaTransformObject tf)
- PetscErrorCode TransformObjectGetName (SalsaTransformObject tf, char ∗∗name)
- PetscErrorCode TransformObjectSetExplanation (SalsaTransformObject tf, char ∗x)
- PetscErrorCode TransformObjectSetSuitabilityFunction (SalsaTransformObject tf, void ∗sctx, PetscErrorCode(∗f)(NumericalProblem, void ∗, SuitabilityValue ∗))
- PetscErrorCode TransformObjectGetSuitabilityFunction (SalsaTransformObject tf, void ∗∗sctx, PetscErrorCode(∗∗f)(NumericalProblem, void ∗, SuitabilityValue ∗))
- PetscErrorCode TransformObjectGetTransformName (SalsaTransformObject tf, char ∗∗name)
- PetscErrorCode TransformObjectGetByName (char ∗trans, char ∗name, SalsaTransformObject ∗tf)
- PetscErrorCode TransformObjectsGetNames (SalsaTransform tf, char ∗∗∗names)
- PetscErrorCode SysProDefineCharAnnotation (char ∗transform, char ∗ann)

- PetscErrorCode TransformCharAnnotationGetIndex (SalsaTransform tf, char ∗ann, int ∗idx, PetscTruth ∗flg)
- PetscErrorCode SysProDefineIntAnnotation (char ∗transform, char ∗ann)
- PetscErrorCode TransformIntAnnotationGetIndex (SalsaTransform tf, char ∗ann, int ∗idx, PetscTruth ∗flg)
- PetscErrorCode TransformObjectMark (SalsaTransformObject tf)
- PetscErrorCode TransformObjectUnmark (SalsaTransformObject tf)
- PetscErrorCode TransformObjectsMarkAll (SalsaTransform tf)
- PetscErrorCode TransformObjectsUnmarkAll (SalsaTransform tf)
- PetscErrorCode TransformObjectGetMark (SalsaTransformObject tf, int ∗m)
- PetscErrorCode TransformGetNUnmarked (SalsaTransform tf, int ∗n)
- PetscErrorCode PreprocessorSaveAprioriSelection (SystemPreprocessor pp)
- PetscErrorCode PreprocessorApplyAprioriSelection (SystemPreprocessor pp)
- PetscErrorCode TransformObjectsUseOnly (SalsaTransform tf, char ∗list)
- PetscErrorCode TransformGetNextUnmarkedItem (SalsaTransform tf, char ∗old, SalsaTransformObject ∗snew, PetscTruth ∗f)
- PetscErrorCode TransformObjectDefineOption (SalsaTransformObject tf, char ∗opt)
- PetscErrorCode TransformObjectAddOption (SalsaTransformObject tf, int v)
- PetscErrorCode TransformObjectAddOptionExplanation (SalsaTransformObject tf, int opt, char ∗ex)
- PetscErrorCode TransformItemOptionsUseOnly (SalsaTransformObject tf, char ∗opt)
- PetscErrorCode TransformItemGetFirstOption (char ∗tf, char ∗it, int ∗v, PetscTruth ∗f)
- PetscErrorCode TransformItemGetNextOption (char ∗tf, char ∗it, int ∗v, PetscTruth ∗f)
- PetscErrorCode TransformItemOptionMark (SalsaTransform tf, char ∗it, int o)
- PetscErrorCode TransformObjectCharAnnotate (SalsaTransformObject tf, char ∗an, char ∗v)
- PetscErrorCode TransformObjectIntAnnotate (SalsaTransformObject tf, char ∗an, int v)
- PetscErrorCode TransformObjectGetIntAnnotation (SalsaTransformObject tf, char ∗an, int ∗v, PetscTruth ∗f)
- PetscErrorCode TransformReportTeXTable (SalsaTransform tf, FILE ∗f)
- PetscErrorCode TransformReportEnabled (SalsaTransform tf, char ∗∗rs)
- PetscErrorCode TransformItemDescribeShort (SalsaTransform tf, char ∗it, int opt, char ∗∗s)
- PetscErrorCode TransformItemDescribeLong (SalsaTransform tf, char ∗it, int opt, char ∗∗s)
- PetscErrorCode TransformSetUserChoices (SalsaTransform tf, PetscTruth ch)
- PetscErrorCode TransformGetUserChoices (SalsaTransform tf, PetscTruth ∗ch)

26.29.1 Define Documentation

26.29.1.1 #define STRDUP(a) ((a) ? strdup(a) : NULL)

Definition at line 504 of file transform.c.

26.29.1.2 #define TFINC 20

Definition at line 8 of file transform.c.

Referenced by NewTransform(), SysProDefineCharAnnotation(), SysProDefineIntAnnotation(), TransformObjectAddOption(), TransformObjectCharAnnotate(), and TransformObjectIntAnnotate().

26.29.2 Function Documentation

26.29.2.1 PetscErrorCode DeregisterTransform (SalsaTransform *tf*)

Definition at line 49 of file transform.c.

References SalsaTransform_::annotations_c, SalsaTransform_::annotations_i, SalsaTransform_::aprioriselection, CHKERRQ(), FreeTransformObject(), ierr, SalsaTransform_::n_objects, and SalsaTransform_::transformobjects.

Referenced by SysProFinalize().

26.29.2.2 PetscErrorCode FreeTransformObject (SalsaTransformObject *tf*)

Definition at line 109 of file transform.c.

References SalsaTransformObject_::annotate_c, SalsaTransformObject_::annotate_-i, CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_-::name, SalsaTransformObject_::optionexplanation, SalsaTransformObject_::options, and SalsaTransformObject_::options_marked.

Referenced by DeregisterTransform().

26.29.2.3 PetscErrorCode NewTransform (char ∗ *name*, SalsaTransform ∗ *tf*)

Define a new class of preprocessors, for instance scaling or permutation.

Definition at line 31 of file transform.c.

References SalsaTransform_::alloc_objects, SalsaTransform_::aprioriselection, CHK-ERRQ(), ierr, SalsaTransform_::n_objects, SalsaTransform_::name, TFINC, and SalsaTransform_::transformobjects.

Referenced by DeclarePreprocessor().

26.29.2.4 PetscErrorCode NewTransformObject (char ∗ *transform*, char ∗ *name*, SalsaTransformObject ∗ *to*)

Create a transform object specified by `name` for the preprocessor class `transform`. The `to` parameter can be NULL if no further specifications of the object are needed, in which case this only registers the name.

Definition at line 92 of file transform.c.

References SalsaTransform_::alloc_objects, CHKERRQ(), ierr, SalsaTransform_::n_-objects, SalsaTransformObject_::name, SalsaTransformObject_::transform, Transfor-mGetByName(), and SalsaTransform_::transformobjects.

Referenced by declareadders(), setup_approximation_choices(), setup_distribution_-choices(), setup_flipsign_choices(), setup_ksp_choices(), setup_pc_choices(), setup_-scaling_choices(), and setup_singleton_choices().

26.29.2.5 PetscErrorCode PreprocessorApplyAprioriSelection (SystemPreprocessor *pp*)

Definition at line 426 of file transform.c.

References SalsaTransform_::aprioriselection, SalsaTransformObject_::marked, SalsaTransform_::n_objects, SystemPreprocessor_::transform, and SalsaTransform_-::transformobjects.

Referenced by PreprocessorSpecificSetup().

26.29.2.6 PetscErrorCode PreprocessorSaveAprioriSelection (SystemPreprocessor *pp*)

Definition at line 414 of file transform.c.

References SalsaTransform_::aprioriselection, SalsaTransformObject_::marked, SalsaTransform_::n_objects, SystemPreprocessor_::transform, and SalsaTransform_-::transformobjects.

Referenced by PreprocessorsOptionsHandling().

26.29.2.7 PetscErrorCode SysProDefineCharAnnotation (char ∗ *transform*, char ∗ *ann*)

Define a character string annotation for a transform. The index of this annotation can be retrieved with TransformCharAnnotationGetIndex(). The actual annotation can be found with TransformItemGetCharAnnotation().

Definition at line 243 of file transform.c.

References SalsaTransform_::annotations_c, CHKERRQ(), ierr, SalsaTransform_::n_-annotate_c, TFINC, and TransformGetByName().

Referenced by setup_ksp_choices().

26.29.2.8 PetscErrorCode SysProDefineIntAnnotation (char ∗ *transform*, char ∗ *ann*)

Define a integer string annotation for a transform. The index of this annotation can be retrieved with TransformIntAnnotationGetIndex(). The actual annotation can be found with TransformItemGetIntAnnotation().

Definition at line 297 of file transform.c.

References SalsaTransform_::annotations_i, CHKERRQ(), ierr, SalsaTransform_::n_-annotate_i, TFINC, and TransformGetByName().

Referenced by setup_distribution_choices(), setup_ksp_choices(), setup_pc_choices(), and setup_scaling_choices().

26.29.2.9 PetscErrorCode TransformCharAnnotationGetIndex (SalsaTransform *tf*, char ∗ *ann*, int ∗ *idx*, PetscTruth ∗ *flg*)

Definition at line 277 of file transform.c.

References SalsaTransform_::annotations_c, CHKERRQ(), ierr, and SalsaTransform_-::n_annotate_c.

Referenced by TransformObjectCharAnnotate().

26.29.2.10 PetscErrorCode TransformGetName (SalsaTransform *tf*, char ∗∗ *name*)

Definition at line 66 of file transform.c.

References SalsaTransform_::name.

26.29.2.11 PetscErrorCode TransformGetNextUnmarkedItem (SalsaTransform *tf*, char ∗ *old*, SalsaTransformObject ∗ *snew*, PetscTruth ∗ *f*)

Find the next unmarked value; if `old` is NULL, the first first unmarked value is given, otherwise the first one after a match with `old`.

Definition at line 477 of file transform.c.

References SalsaTransformObject_::marked, SalsaTransform_::n_objects, SalsaTransformObject_::name, SalsaTransform_::transformobjects, and TRUTH.

Referenced by ChooseFirstTransform(), PreprocessedSolution(), and Preprocessor-sOptionsHandling().

26.29.2.12 PetscErrorCode TransformGetNUnmarked (SalsaTransform *tf*, int ∗ *n*)

Definition at line 402 of file transform.c.

References SalsaTransformObject_::marked, SalsaTransform_::n_objects, and SalsaTransform_::transformobjects.

Referenced by PreprocessorsOptionsHandling().

26.29.2.13 PetscErrorCode TransformGetObjects (SalsaTransform *tf*, int ∗ *n*, SalsaTransformObject ∗∗ *objs*)

Definition at line 76 of file transform.c.

References SalsaTransform_::n_objects, and SalsaTransform_::transformobjects.

Referenced by disable_ksps(), specific_distribution_choices(), and specific_scaling_-choices().

26.29.2.14 PetscErrorCode TransformGetUserChoices (SalsaTransform *tf*, PetscTruth ∗ *ch*)

Definition at line 794 of file transform.c.

References SalsaTransform_::userchoices.

Referenced by PreprocessedSolution().

26.29.2.15 PetscErrorCode TransformIntAnnotationGetIndex (SalsaTransform *tf*, char ∗ *ann*, int ∗ *idx*, PetscTruth ∗ *flg*)

Definition at line 329 of file transform.c.

References SalsaTransform_::annotations_i, CHKERRQ(), ierr, and SalsaTransform_-::n_annotate_i.

Referenced by TransformObjectIntAnnotate().

26.29.2.16 PetscErrorCode TransformItemDescribeLong (SalsaTransform *tf*, char ∗ *it*, int *opt*, char ∗∗ *s*)

Definition at line 755 of file transform.c.

References CHKERRQ(), SalsaTransformObject_::explanation, ierr, SalsaTransform_::name, SalsaTransformObject_::options, and TransformObject-GetByName().

26.29.2.17 PetscErrorCode TransformItemDescribeShort (SalsaTransform *tf*, char ∗ *it*, int *opt*, char ∗∗ *s*)

Definition at line 737 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_::name, SalsaTransform_::name, and TransformObject-GetByName().

26.29.2.18 PetscErrorCode TransformItemGetFirstOption (char ∗ *tf*, char ∗ *it*, int ∗ *v*, PetscTruth ∗ *f*)

Definition at line 572 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_::options, SalsaTransformObject_::options_marked, and TransformObjectGetByName().

Referenced by PreprocessedSolution().

26.29.2.19 PetscErrorCode TransformItemGetNextOption (char ∗ *tf*, char ∗ *it*, int ∗ *v*, PetscTruth ∗ *f*)

Definition at line 595 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_::options, SalsaTransformObject_::options_marked, and TransformObjectGetByName().

Referenced by PreprocessedSolution().

26.29.2.20 PetscErrorCode TransformItemOptionMark (SalsaTransform *tf*, char ∗ *it*, int *o*)

Definition at line 619 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::marked, SalsaTransformObject_::n_options, SalsaTransform_::name, SalsaTransformObject_-::options, SalsaTransformObject_::options_marked, and TransformObjectGetBy-Name().

26.29.2.21 PetscErrorCode TransformItemOptionsUseOnly (SalsaTransformObject *tf*, char ∗ *opt*)

Definition at line 555 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject_::n_options, and TransformOb-jectAddOption().

Referenced by PreprocessorsOptionsHandling().

26.29.2.22 PetscErrorCode TransformObjectAddOption (SalsaTransformObject *tf*, int *v*)

Definition at line 517 of file transform.c.

References SalsaTransformObject_::alloc_options, CHKERRQ(), ierr, SalsaTransformObject_::n_options, SalsaTransformObject_::name, SalsaTransformObject_::optionexplanation, SalsaTransformObject_::options, SalsaTransformObject_::options_marked, and TFINC.

Referenced by declareadders(), setup_ksp_choices(), setup_pc_choices(), and TransformItemOptionsUseOnly().

26.29.2.23 PetscErrorCode TransformObjectAddOptionExplanation (SalsaTransformObject *tf*, int *opt*, char ∗ *ex*)

Definition at line 539 of file transform.c.

References SalsaTransformObject_::n_options, SalsaTransformObject_-::optionexplanation, and SalsaTransformObject_::options.

Referenced by setup_pc_choices().

26.29.2.24 PetscErrorCode TransformObjectCharAnnotate (SalsaTransformObject *tf*, char ∗ *an*, char ∗ *v*)

Definition at line 634 of file transform.c.

References SalsaTransformObject_::alloc_annotate_c, SalsaTransformObject_-::annotate_c, CHKERRQ(), ierr, TFINC, SalsaTransformObject_::transform, and TransformCharAnnotationGetIndex().

26.29.2.25 PetscErrorCode TransformObjectDefineOption (SalsaTransformObject *tf*, char ∗ *opt*)

Definition at line 508 of file transform.c.

References SalsaTransformObject_::option.

Referenced by declareadders(), setup_ksp_choices(), and setup_pc_choices().

26.29.2.26 PetscErrorCode TransformObjectGetByName (char ∗ *trans*, char ∗ *name*, SalsaTransformObject ∗ *tf*)

Definition at line 202 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, SalsaTransformObject_-::name, TransformGetByName(), and SalsaTransform_::transformobjects.

Referenced by disable_ksps(), disable_pcs(), is_gmres_method(), pcoptionshandling(), setup_ksp(), specific_approximation_choices(), specific_flipsign_choices(), specific_-singleton_choices(), TransformItemDescribeLong(), TransformItemDescribeShort(), TransformItemGetFirstOption(), TransformItemGetNextOption(), TransformItemOptionMark(), TransformObjectsUseOnly(), and unset_ksps().

26.29.2.27 PetscErrorCode TransformObjectGetIntAnnotation (SalsaTransformObject *tf*, char ∗ *an*, int ∗ *v*, PetscTruth ∗ *f*)

Definition at line 676 of file transform.c.

References SalsaTransformObject_::annotate_i, SalsaTransform_::annotations_i, SalsaTransform_::n_annotate_i, SalsaTransformObject_::transform, and TRUTH.

Referenced by disable_ksps(), is_gmres_method(), pcoptionshandling(), set_ksp_-options(), specific_distribution_choices(), and specific_scaling_choices().

26.29.2.28 PetscErrorCode TransformObjectGetMark (SalsaTransformObject *tf*, int ∗ *m*)

Definition at line 393 of file transform.c.

References SalsaTransformObject_::marked.

Referenced by TransformReportEnabled().

26.29.2.29 PetscErrorCode TransformObjectGetName (SalsaTransformObject *tf*, char ∗∗ *name*)

Definition at line 130 of file transform.c.

References SalsaTransformObject_::name.

Referenced by ChooseFirstTransform(), disable_ksps(), PreprocessedSolution(), PreprocessorsOptionsHandling(), and set_ksp_options().

26.29.2.30 PetscErrorCode TransformObjectGetSuitabilityFunction (SalsaTransformObject *tf*, void ∗∗ *sctx*, PetscErrorCode(∗∗)(NumericalProblem, void ∗, SuitabilityValue ∗) *f*)

Retrieve the suitability function and context; see Suitability functions. Both arguments can be null.

Definition at line 182 of file transform.c.

References SalsaTransformObject_::suitabilityctx, and SalsaTransformObject_::suitabilityfunction.

Referenced by PreprocessorSpecificSetup(), and unset_ksps().

26.29.2.31 PetscErrorCode TransformObjectGetTransformName (SalsaTransformObject *tf*, char ∗∗ *name*)

Definition at line 192 of file transform.c.

References SalsaTransform_::name, and SalsaTransformObject_::transform.

26.29.2.32 PetscErrorCode TransformObjectIntAnnotate (SalsaTransformObject *tf*, char ∗ *an*, int *v*)

Definition at line 655 of file transform.c.

References SalsaTransformObject_::alloc_annotate_i, SalsaTransformObject_::annotate_i, CHKERRQ(), ierr, TFINC, SalsaTransformObject_::transform, and TransformIntAnnotationGetIndex().

Referenced by setup_distribution_choices(), setup_ksp_choices(), setup_pc_choices(), and setup_scaling_choices().

26.29.2.33 PetscErrorCode TransformObjectMark (SalsaTransformObject *tf*)

Definition at line 345 of file transform.c.

References SalsaTransformObject_::marked, SalsaTransformObject_::n_options, and SalsaTransformObject_::options_marked.

Referenced by disable_ksps(), disable_pcs(), pcoptionshandling(), Preprocessor-SpecificSetup(), specific_approximation_choices(), specific_distribution_choices(), specific_flipsign_choices(), specific_scaling_choices(), specific_singleton_choices(), TransformObjectsMarkAll(), and TransformObjectsUseOnly().

26.29.2.34 PetscErrorCode TransformObjectSetExplanation (SalsaTransformObject *tf*, char ∗ *x*)

Definition at line 139 of file transform.c.

References SalsaTransformObject_::explanation.

Referenced by setup_approximation_choices(), setup_distribution_choices(), setup_-flipsign_choices(), setup_ksp_choices(), setup_pc_choices(), setup_scaling_choices(), and setup_singleton_choices().

26.29.2.35 PetscErrorCode TransformObjectSetSuitabilityFunction (SalsaTransformObject *tf*, void ∗ *sctx*, PetscErrorCode(∗)(NumericalProblem, void ∗, SuitabilityValue ∗) *f*)

Set the suitability function; see Suitability functions

Definition at line 168 of file transform.c.

References SalsaTransformObject_::suitabilityctx, and SalsaTransformObject_-::suitabilityfunction.

Referenced by setup_ksp_choices().

26.29.2.36 PetscErrorCode TransformObjectsGetNames (SalsaTransform *tf*, char ∗∗∗ *names*)

Get the names of all declared transformobjects. An array is allocated for the names, which needs to be PetscFree()'d.

Definition at line 223 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, SalsaTransformObject_-::name, and SalsaTransform_::transformobjects.

Referenced by ContinueRetrievingAllPreprocessors(), and TransformReportEnabled().

26.29.2.37 PetscErrorCode TransformObjectsMarkAll (SalsaTransform *tf*)

Definition at line 369 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, TransformObjectMark(), and SalsaTransform_::transformobjects.

Referenced by TransformObjectsUseOnly().

26.29.2.38 PetscErrorCode TransformObjectsUnmarkAll (SalsaTransform *tf*)

Definition at line 381 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, SalsaTransform_::transformobjects, and TransformObjectUnmark().

Referenced by disable_ksps(), and TransformObjectsUseOnly().

26.29.2.39 PetscErrorCode TransformObjectsUseOnly (SalsaTransform *tf*, char ∗ *list*)

Mark a list of names as to be used.

Cases:

- `"name1,name2,name3"` : all other names are marked as not to be used
- `"not,name1,name2"` : all names will be used, except for the ones listed

Definition at line 445 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::name, TransformObjectGetByName(), TransformObjectMark(), TransformObjectsMarkAll(), TransformObjectsUnmarkAll(), and TransformObjectUnmark().

Referenced by PreprocessorsOptionsHandling().

26.29.2.40 PetscErrorCode TransformObjectUnmark (SalsaTransformObject *tf*)

Definition at line 357 of file transform.c.

References SalsaTransformObject_::marked, SalsaTransformObject_::n_options, and SalsaTransformObject_::options_marked.

Referenced by TransformObjectsUnmarkAll(), and TransformObjectsUseOnly().

26.29.2.41 PetscErrorCode TransformReportEnabled (SalsaTransform *tf*, char $**$ *rs*)

Definition at line 715 of file transform.c.

References CHKERRQ(), ierr, SalsaTransform_::n_objects, TransformObjectGet-Mark(), SalsaTransform_::transformobjects, and TransformObjectsGetNames().

Referenced by ReportEnabledPreprocessors().

26.29.2.42 PetscErrorCode TransformReportTeXTable (SalsaTransform *tf*, FILE $*$ *f*)

Definition at line 693 of file transform.c.

References SalsaTransformObject_::explanation, SalsaTransform_::n_-objects, SalsaTransformObject_::n_options, SalsaTransformObject_::name, SalsaTransformObject_::options, and SalsaTransform_::transformobjects.

26.29.2.43 PetscErrorCode TransformSetUserChoices (SalsaTransform *tf*, PetscTruth *ch*)

Definition at line 785 of file transform.c.

References SalsaTransform_::userchoices.

Referenced by PreprocessorsOptionsHandling().

26.30 u1.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
```

Functions

- static PetscErrorCode copy (NumericalProblem problem, void ∗dum, NumericalSolution ∗rsol)
- int main (int argc, char ∗∗argv)

26.30.1 Function Documentation

26.30.1.1 static PetscErrorCode copy (NumericalProblem *problem*, void ∗ *dum*, NumericalSolution ∗ *rsol*) `[static]`

Definition at line 7 of file u1.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.30.1.2 int main (int *argc*, char ∗∗ *argv*)

Definition at line 17 of file u1.c.

References CHKERRQ(), copy(), ierr, SysProDeclareFunctions(), SysProFinalize(), and SysProInitialize().

26.31 u12.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
#include "sysprolinear.h"
#include "testmat.c"
```

Functions

- static PetscErrorCode solvelinear (NumericalProblem problem, void ∗dum, NumericalSolution ∗rsol)
- int main (int argc, char ∗∗argv)

26.31.1 Function Documentation

26.31.1.1 int main (int *argc*, char ∗∗ *argv*)

Definition at line 28 of file u12.c.

References CHKERRQ(), CreateLinearSystem(), ierr, LinearSystemSetParts(), PreprocessedProblemSolving(), solvelinear(), SysProDeclareFunctions(), SysProFinalize(), and SysProInitialize().

26.31.1.2 static PetscErrorCode solvelinear (NumericalProblem *problem*, void ∗ *dum*, NumericalSolution ∗ *rsol*) `[static]`

Definition at line 13 of file u12.c.

References CHKERRQ(), ierr, and LinearSystemGetParts().

Referenced by main().

26.32 u13.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
#include "sysprotransform.h"
#include "sysprolinear.h"
#include "testmat.c"
```

Functions

- static PetscErrorCode create_solver (NumericalProblem prob, void ∗∗ctx)
- static PetscErrorCode destroy_solver (void ∗ctx)
- static PetscErrorCode setup_pc_choices ()

- static PetscErrorCode setup_pc (char ∗type, int pcv, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode solvelinear (NumericalProblem problem, void ∗dum, NumericalSolution ∗rsol)
- static PetscErrorCode destroysolution (NumericalSolution sol)
- int main (int argc, char ∗∗argv)

26.32.1 Function Documentation

26.32.1.1 static PetscErrorCode create_solver (NumericalProblem *prob*, void ∗∗ *ctx*) `[static]`

Create a solver and install a monitor that dynamically increases the maximum number of iterations.

Definition at line 20 of file u13.c.

References CHKERRQ(), ierr, and NumericalProblemGetComm().

Referenced by main().

26.32.1.2 static PetscErrorCode destroy_solver (void ∗ *ctx*) `[static]`

Definition at line 32 of file u13.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.32.1.3 static PetscErrorCode destroysolution (NumericalSolution *sol*) `[static]`

Definition at line 124 of file u13.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.32.1.4 int main (int *argc*, char ∗∗ *argv*)

Definition at line 134 of file u13.c.

References CHKERRQ(), create_solver(), CreateLinearSystem(), DeclarePreprocessor(), destroy_solver(), destroysolution(), ierr, LinearSystemSetParts(), PreprocessedProblemSolving(), PreprocessorsOptionsHandling(), setup_pc(), setup_pc_choices(), solvelinear(), SysProDeclareFunctions(), SysProDeclareTraceFunction(), SysProDefaultTrace(), SysProFinalize(), and SysProInitialize().

26.32.1.5 static PetscErrorCode setup_pc (char ∗ *type*, int *pcv*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem ∗ *outproblem*, void ∗ *gctx*, void ∗∗ *ctx*, PetscTruth ∗ *success*) `[static]`

Definition at line 56 of file u13.c.

References CHKERRQ(), ierr, and LinearSystemGetParts().

Referenced by main().

26.32.1.6 static PetscErrorCode setup_pc_choices () `[static]`

Definition at line 43 of file u13.c.

References CHKERRQ(), ierr, and NewTransformObject().

Referenced by main().

26.32.1.7 static PetscErrorCode solvelinear (NumericalProblem *problem*, void ∗ *dum*, NumericalSolution ∗ *rsol*) `[static]`

Definition at line 102 of file u13.c.

References CHKERRQ(), ierr, and LinearSystemGetParts().

Referenced by main().

26.33 u14.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
#include "sysprotransform.h"
```

```
#include "sysprolinear.h"
#include "testmat.c"
```

Functions

- static PetscErrorCode create_solver (NumericalProblem prob, void ∗∗ctx)
- static PetscErrorCode destroy_solver (void ∗ctx)
- static PetscErrorCode setup_pc_choices ()
- static PetscErrorCode setup_pc (char ∗type, int pcv, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode unset_pc (char ∗type, PetscTruth overwrite, void ∗ctx, void ∗gctx, NumericalProblem thisproblem, NumericalProblem upproblem, NumericalSolution old, NumericalSolution nnew)
- static PetscErrorCode solvelinear (NumericalProblem problem, void ∗dum, NumericalSolution ∗rsol)
- int main (int argc, char ∗∗argv)

26.33.1 Function Documentation

26.33.1.1 static PetscErrorCode create_solver (NumericalProblem *prob*, void ∗∗ *ctx*) `[static]`

Create a solver and install a monitor that dynamically increases the maximum number of iterations.

Definition at line 20 of file u14.c.

References CHKERRQ(), ierr, and NumericalProblemGetComm().

Referenced by main().

26.33.1.2 static PetscErrorCode destroy_solver (void ∗ *ctx*) `[static]`

Definition at line 32 of file u14.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.33.1.3 int main (int *argc*, char ∗∗ *argv*)

Definition at line 128 of file u14.c.

References CHKERRQ(), create_solver(), CreateLinearSystem(), DeclarePreprocessor(), DeclareScalingPreprocessor(), destroy_solver(), ierr, LinearCreateNumericalSolution(), LinearDeleteNumericalSolution(), LinearSolutionGetVector(), LinearSystemSetParts(), PreprocessedProblemSolving(), PreprocessorsOptionsHandling(), setup_pc(), setup_pc_choices(), solvelinear(), SysProDeclareFunctions(), SysProDeclareTraceFunction(), SysProDefaultTrace(), SysProFinalize(), and SysProInitialize().

26.33.1.4 static PetscErrorCode setup_pc (char ∗ *type*, int *pcv*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem ∗ *outproblem*, void ∗ *gctx*, void ∗∗ *ctx*, PetscTruth ∗ *success*) `[static]`

Definition at line 56 of file u14.c.

References CHKERRQ(), ierr, and LinearSystemGetParts().

Referenced by main().

26.33.1.5 static PetscErrorCode setup_pc_choices () `[static]`

Definition at line 43 of file u14.c.

References CHKERRQ(), ierr, and NewTransformObject().

Referenced by main().

26.33.1.6 static PetscErrorCode solvelinear (NumericalProblem *problem*, void ∗ *dum*, NumericalSolution ∗ *rsol*) `[static]`

Definition at line 107 of file u14.c.

References CHKERRQ(), ierr, LinearCreateNumericalSolution(), LinearSolutionSetVector(), and LinearSystemGetParts().

Referenced by main().

26.33.1.7 static PetscErrorCode unset_pc (char ∗ *type*, PetscTruth *overwrite*, void ∗ *ctx*, void ∗ *gctx*, NumericalProblem *thisproblem*, NumericalProblem *upproblem*, NumericalSolution *old*, NumericalSolution *nnew*) `[static]`

Definition at line 84 of file u14.c.

References CHKERRQ(), ierr, and LinearSolutionCopy().

26.34 u15.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
#include "anamodsalsamodules.h"
#include "sysprotransform.h"
#include "sysprolinear.h"
#include "anamod.h"
#include "nmd.h"
#include "testmat.c"
```

Functions

- static PetscErrorCode create_solver (NumericalProblem prob, void ∗∗ctx)
- static PetscErrorCode destroy_solver (void ∗ctx)
- static PetscErrorCode setup_pc_choices ()
- static PetscErrorCode setup_pc (char ∗type, int pcv, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode unset_pc (char ∗type, PetscTruth overwrite, void ∗ctx, void ∗gctx, NumericalProblem thisproblem, NumericalProblem upproblem, NumericalSolution old, NumericalSolution nnew)
- static PetscErrorCode solvelinear (NumericalProblem problem, void ∗dum, NumericalSolution ∗rsol)
- int main (int argc, char ∗∗argv)

26.34.1 Function Documentation

26.34.1.1 static PetscErrorCode create_solver (NumericalProblem *prob*, void ∗∗ *ctx*) `[static]`

Create a solver and install a monitor that dynamically increases the maximum number of iterations.

Definition at line 23 of file u15.c.

References CHKERRQ(), ierr, and NumericalProblemGetComm().

Referenced by main().

26.34.1.2 static PetscErrorCode destroy_solver (void ∗ *ctx*) `[static]`

Definition at line 35 of file u15.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.34.1.3 int main (int *argc*, char ∗∗ *argv*)

Definition at line 131 of file u15.c.

References CHKERRQ(), create_solver(), CreateLinearSystem(), DeclarePreprocessor(), DeclareSingletonPreprocessor(), destroy_solver(), ierr, LinearCreateNumericalSolution(), LinearDeleteNumericalSolution(), LinearSolutionGetVector(), LinearSystemGetMetadata(), LinearSystemSetMetadata(), LinearSystemSetParts(), PreprocessedProblemSolving(), PreprocessorsOptionsHandling(), setup_pc(), setup_pc_-choices(), solvelinear(), SysProDeclareFunctions(), SysProDeclareTraceFunction(), SysProDefaultTrace(), SysProFinalize(), and SysProInitialize().

26.34.1.4 static PetscErrorCode setup_pc (char ∗ *type*, int *pcv*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem ∗ *outproblem*, void ∗ *gctx*, void ∗∗ *ctx*, PetscTruth ∗ *success*) `[static]`

Definition at line 59 of file u15.c.

References CHKERRQ(), ierr, and LinearSystemGetParts().

Referenced by main().

26.34.1.5 static PetscErrorCode setup_pc_choices () `[static]`

Definition at line 46 of file u15.c.

References CHKERRQ(), ierr, and NewTransformObject().

Referenced by main().

26.34.1.6 static PetscErrorCode solvelinear (NumericalProblem *problem*, void ∗ *dum*, NumericalSolution ∗ *rsol*) `[static]`

Definition at line 110 of file u15.c.

References CHKERRQ(), ierr, LinearCreateNumericalSolution(), LinearSolution-SetVector(), and LinearSystemGetParts().

Referenced by main().

26.34.1.7 static PetscErrorCode unset_pc (char ∗ *type*, PetscTruth *overwrite*, void ∗ *ctx*, void ∗ *gctx*, NumericalProblem *thisproblem*, NumericalProblem *upproblem*, NumericalSolution *old*, NumericalSolution *nnew*) `[static]`

Definition at line 87 of file u15.c.

References CHKERRQ(), ierr, and LinearSolutionCopy().

26.35 u16.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
#include "anamodsalsamodules.h"
#include "sysprotransform.h"
#include "sysprolinear.h"
#include "anamod.h"
```

```
#include "nmd.h"
#include "testmat16.c"
```

Functions

- static PetscErrorCode create_solver (NumericalProblem prob, void ∗∗ctx)
- static PetscErrorCode destroy_solver (void ∗ctx)
- static PetscErrorCode setup_pc_choices ()
- static PetscErrorCode setup_pc (char ∗type, int pcv, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem ∗outproblem, void ∗gctx, void ∗∗ctx, PetscTruth ∗success)
- static PetscErrorCode unset_pc (char ∗type, PetscTruth overwrite, void ∗ctx, void ∗gctx, NumericalProblem thisproblem, NumericalProblem upproblem, NumericalSolution old, NumericalSolution new)
- static PetscErrorCode solvelinear (NumericalProblem problem, void ∗dum, NumericalSolution ∗rsol)
- int main (int argc, char ∗∗argv)

26.35.1 Function Documentation

26.35.1.1 static PetscErrorCode create_solver (NumericalProblem *prob*, void ∗∗ *ctx*) `[static]`

Create a solver and install a monitor that dynamically increases the maximum number of iterations.

Definition at line 23 of file u16.c.

References CHKERRQ(), ierr, and NumericalProblemGetComm().

Referenced by main().

26.35.1.2 static PetscErrorCode destroy_solver (void ∗ *ctx*) `[static]`

Definition at line 35 of file u16.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.35.1.3 int main (int *argc*, char ∗∗ *argv*)

Definition at line 131 of file u16.c.

References CHKERRQ(), create_solver(), CreateLinearSystem(), DeclarePreprocessor(), DeclareSingletonPreprocessor(), destroy_solver(), ierr, LinearCreateNumericalSolution(), LinearDeleteNumericalSolution(), LinearSolutionGetVector(), LinearSystemGetMetadata(), LinearSystemSetMetadata(), LinearSystemSetParts(), PreprocessedProblemSolving(), PreprocessorsOptionsHandling(), setup_pc(), setup_pc_-choices(), solvelinear(), SysProDeclareFunctions(), SysProDeclareTraceFunction(), SysProDefaultTrace(), SysProFinalize(), and SysProInitialize().

26.35.1.4 static PetscErrorCode setup_pc (char ∗ *type*, int *pcv*, PetscTruth *overwrite*, NumericalProblem *inproblem*, NumericalProblem ∗ *outproblem*, void ∗ *gctx*, void ∗∗ *ctx*, PetscTruth ∗ *success*) `[static]`

Definition at line 59 of file u16.c.

References CHKERRQ(), ierr, and LinearSystemGetParts().

Referenced by main().

26.35.1.5 static PetscErrorCode setup_pc_choices () `[static]`

Definition at line 46 of file u16.c.

References CHKERRQ(), ierr, and NewTransformObject().

Referenced by main().

26.35.1.6 static PetscErrorCode solvelinear (NumericalProblem *problem*, void ∗ *dum*, NumericalSolution ∗ *rsol*) `[static]`

Definition at line 110 of file u16.c.

References CHKERRQ(), ierr, LinearCreateNumericalSolution(), LinearSolutionSetVector(), and LinearSystemGetParts().

Referenced by main().

26.35.1.7 static PetscErrorCode unset_pc (char ∗ *type*, PetscTruth *overwrite*, void ∗ *ctx*, void ∗ *gctx*, NumericalProblem *thisproblem*, NumericalProblem *upproblem*, NumericalSolution *old*, NumericalSolution *new*) `[static]`

Definition at line 87 of file u16.c.

References CHKERRQ(), ierr, and LinearSolutionCopy().

26.36 u2.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
```

Functions

- static PetscErrorCode solvebycopy (NumericalProblem problem, void ∗dum, NumericalSolution ∗rsol)
- int main (int argc, char ∗∗argv)

26.36.1 Function Documentation

26.36.1.1 int main (int *argc*, char ∗∗ *argv*)

Definition at line 19 of file u2.c.

References CHKERRQ(), ierr, PreprocessedProblemSolving(), solvebycopy(), SysProDeclareFunctions(), SysProFinalize(), and SysProInitialize().

26.36.1.2 static PetscErrorCode solvebycopy (NumericalProblem *problem*, void ∗ *dum*, NumericalSolution ∗ *rsol*) `[static]`

Definition at line 9 of file u2.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.37 u3.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
#include "sysprotransform.h"
#include "string.h"
```

Functions

- static PetscErrorCode solvebycopy (NumericalProblem problem, void ∗dum, NumericalSolution ∗rsol)
- static PetscErrorCode delprob (NumericalProblem p)
- static PetscErrorCode makesol (NumericalProblem p, NumericalSolution ∗rs)
- static PetscErrorCode delsol (NumericalSolution s)
- static PetscErrorCode adder (char ∗choice, int optionvalue, PetscTruth overwrite, NumericalProblem oldproblem, NumericalProblem ∗rnew, void ∗ctx, void ∗∗lctx, PetscTruth ∗success)
- static PetscErrorCode unadder (char ∗choice, PetscTruth overwrite, void ∗ctx, void ∗lctx, NumericalProblem pproblem, NumericalProblem oproblem, NumericalSolution psol, NumericalSolution osol)
- static PetscErrorCode declareadders ()
- int main (int argc, char ∗∗argv)

26.37.1 Function Documentation

26.37.1.1 static PetscErrorCode adder (char ∗ *choice*, int *optionvalue*, PetscTruth *overwrite*, NumericalProblem *oldproblem*, NumericalProblem ∗ *rnew*, void ∗ *ctx*, void ∗∗ *lctx*, PetscTruth ∗ *success*) `[static]`

Definition at line 63 of file u3.c.

References CHKERRQ(), ierr, and SysProTraceMessage().

Referenced by main().

26.37.1.2 static PetscErrorCode declareadders () `[static]`

Definition at line 97 of file u3.c.

References CHKERRQ(), ierr, and NewTransformObject().

Referenced by main().

26.37.1.3 static PetscErrorCode delprob (NumericalProblem *p*) `[static]`

Definition at line 28 of file u3.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.37.1.4 static PetscErrorCode delsol (NumericalSolution *s*) `[static]`

Definition at line 51 of file u3.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.37.1.5 int main (int *argc*, char ∗∗ *argv*)

Definition at line 107 of file u3.c.

References adder(), CHKERRQ(), declareadders(), DeclarePreprocessor(), delprob(), delsol(), ierr, makesol(), PreprocessedProblemSolving(), solvebycopy(), SysProDeclareFunctions(), SysProDeclareTraceFunction(), SysProDefaultTrace(), SysProFinalize(), SysProInitialize(), and unadder().

26.37.1.6 static PetscErrorCode makesol (NumericalProblem *p*, NumericalSolution ∗ *rs*) `[static]`

Definition at line 39 of file u3.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.37.1.7 static PetscErrorCode solvebycopy (NumericalProblem *problem*, void ∗ *dum*, NumericalSolution ∗ *rsol*) `[static]`

Definition at line 14 of file u3.c.

References CHKERRQ(), ierr, and SysProTraceMessage().

Referenced by main().

26.37.1.8 static PetscErrorCode unadder (char ∗ *choice*, PetscTruth *overwrite*, void ∗ *ctx*, void ∗ *lctx*, NumericalProblem *pproblem*, NumericalProblem *oproblem*, NumericalSolution *psol*, NumericalSolution *osol*) `[static]`

Definition at line 84 of file u3.c.

Referenced by main().

26.38 u4.c File Reference

```
#include <stdlib.h>
#include "syspro.h"
#include "sysprotransform.h"
#include "string.h"
```

Functions

- static PetscErrorCode solvebycopy (NumericalProblem problem, void ∗dum, NumericalSolution ∗rsol)
- static PetscErrorCode makeintctx (NumericalProblem problem, void ∗∗ctx)
- static PetscErrorCode delintctx (void ∗ctx)
- static PetscErrorCode delprob (NumericalProblem p)
- static PetscErrorCode makesol (NumericalProblem p, NumericalSolution ∗rs)
- static PetscErrorCode delsol (NumericalSolution s)
- static PetscErrorCode adder (char ∗choice, int optionvalue, PetscTruth overwrite, NumericalProblem oldproblem, NumericalProblem ∗rnew, void ∗ctx, void ∗∗lctx, PetscTruth ∗success)
- static PetscErrorCode unadder (char ∗choice, PetscTruth overwrite, void ∗ctx, void ∗lctx, NumericalProblem pproblem, NumericalProblem oproblem, NumericalSolution psol, NumericalSolution osol)

- static PetscErrorCode declareadders ()
- int main (int argc, char $**$argv)

26.38.1 Function Documentation

26.38.1.1 static PetscErrorCode adder (char $*$ *choice*, int *optionvalue*, PetscTruth *overwrite*, NumericalProblem *oldproblem*, NumericalProblem $*$ *rnew*, void $*$ *ctx*, void $**$ *lctx*, PetscTruth $*$ *success*) `[static]`

Definition at line 88 of file u4.c.

References CHKERRQ(), ierr, and SysProTraceMessage().

Referenced by main().

26.38.1.2 static PetscErrorCode declareadders () `[static]`

Definition at line 123 of file u4.c.

References CHKERRQ(), ierr, NewTransformObject(), TransformObjectAddOption(), and TransformObjectDefineOption().

Referenced by main().

26.38.1.3 static PetscErrorCode delintctx (void $*$ *ctx*) `[static]`

Definition at line 41 of file u4.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.38.1.4 static PetscErrorCode delprob (NumericalProblem *p*) `[static]`

Definition at line 52 of file u4.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.38.1.5 static PetscErrorCode delsol (NumericalSolution *s*) `[static]`

Definition at line 75 of file u4.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.38.1.6 int main (int *argc*, char ∗∗ *argv*)

Definition at line 137 of file u4.c.

References adder(), CHKERRQ(), declareadders(), DeclarePreprocessor(), delintctx(), delprob(), delsol(), ierr, makeintctx(), makesol(), PreprocessedProblemSolving(), PreprocessorsOptionsHandling(), solvebycopy(), SysProDeclareFunctions(), SysProDeclareTraceFunction(), SysProDefaultTrace(), SysProFinalize(), SysProInitialize(), and unadder().

26.38.1.7 static PetscErrorCode makeintctx (NumericalProblem *problem*, void ∗∗ *ctx*) `[static]`

Definition at line 30 of file u4.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.38.1.8 static PetscErrorCode makesol (NumericalProblem *p*, NumericalSolution ∗ *rs*) `[static]`

Definition at line 63 of file u4.c.

References CHKERRQ(), and ierr.

Referenced by main().

26.38.1.9 static PetscErrorCode solvebycopy (NumericalProblem *problem*, void ∗ *dum*, NumericalSolution ∗ *rsol*) `[static]`

Definition at line 16 of file u4.c.

References CHKERRQ(), ierr, and SysProTraceMessage().

Referenced by main().

26.38.1.10 static PetscErrorCode unadder (char * *choice*, PetscTruth *overwrite*, void * *ctx*, void * *lctx*, NumericalProblem *pproblem*, NumericalProblem *oproblem*, NumericalSolution *psol*, NumericalSolution *osol*) `[static]`

Definition at line 110 of file u4.c.

Referenced by main().

Index