

## System Preprocessors

Generated by Doxygen 1.5.9

Sun Oct 4 04:03:04 2009

## Contents

<b>1</b>	<b>SysPro: a System preProcessors library</b>	<b>1</b>
1.1	Introduction . . . . .	1
1.2	Topics . . . . .	2
1.3	Change log . . . . .	2
<b>2</b>	<b>Approximate the coefficient matrix</b>	<b>2</b>
<b>3</b>	<b>Permutation / load balancing</b>	<b>3</b>
<b>4</b>	<b>Flip the sign of a matrix</b>	<b>3</b>
<b>5</b>	<b>The iterative method</b>	<b>3</b>
<b>6</b>	<b>The Linear package for SysPro</b>	<b>3</b>
<b>7</b>	<b>Transformations for linear systems</b>	<b>4</b>
<b>8</b>	<b>Definition of a linear system</b>	<b>4</b>
<b>9</b>	<b>linearsolution</b>	<b>4</b>
<b>10</b>	<b>Solution Statistics</b>	<b>5</b>
<b>11</b>	<b>The linear solution context</b>	<b>5</b>
<b>12</b>	<b>Command line options handling</b>	<b>5</b>
<b>13</b>	<b>The preconditioner</b>	<b>6</b>
<b>14</b>	<b>Use of the SysPro package</b>	<b>6</b>
14.1	Setup of the SysPro system . . . . .	6
14.1.1	Global declarations . . . . .	6
14.1.2	Preprocessor declaration . . . . .	6
14.2	Setup and invocation . . . . .	7
14.3	Usage modes . . . . .	7

<b>15 The interface to other packages</b>	<b>7</b>
15.1 Computational modules interface . . . . .	7
<b>16 Context handling</b>	<b>8</b>
16.1 contexts . . . . .	8
16.2 contexts . . . . .	8
<b>17 Preprocessor reporting</b>	<b>8</b>
<b>18 Scale a linear system</b>	<b>9</b>
<b>19 Eliminate fully determined (singleton) rows from a matrix</b>	<b>9</b>
<b>20 Tracing the preprocessors</b>	<b>9</b>
<b>21 Preprocessor definition</b>	<b>10</b>
21.1 Class definition . . . . .	10
21.2 Individual preprocessor definition . . . . .	10
<b>22 Suitability functions</b>	<b>10</b>
<b>23 Data Structure Index</b>	<b>10</b>
23.1 Data Structures . . . . .	10
<b>24 File Index</b>	<b>11</b>
24.1 File List . . . . .	11
<b>25 Data Structure Documentation</b>	<b>12</b>
25.1 LinearSolution_ Struct Reference . . . . .	12
25.1.1 Detailed Description . . . . .	13
25.1.2 Field Documentation . . . . .	13
25.2 LinearSystem_ Struct Reference . . . . .	14
25.2.1 Detailed Description . . . . .	14
25.2.2 Field Documentation . . . . .	14
25.3 NumericalProblem_ Struct Reference . . . . .	17

25.3.1 Detailed Description . . . . .	17
25.3.2 Field Documentation . . . . .	17
25.4 PreprocessorsGlobalInfo_ Struct Reference . . . . .	18
25.4.1 Detailed Description . . . . .	18
25.4.2 Field Documentation . . . . .	18
25.5 SalsaTransform_ Struct Reference . . . . .	21
25.5.1 Detailed Description . . . . .	22
25.5.2 Field Documentation . . . . .	22
25.6 SalsaTransformObject_ Struct Reference . . . . .	24
25.6.1 Detailed Description . . . . .	25
25.6.2 Field Documentation . . . . .	25
25.7 singleton_struct Struct Reference . . . . .	28
25.7.1 Detailed Description . . . . .	28
25.7.2 Field Documentation . . . . .	29
25.8 SystemPreprocessor_ Struct Reference . . . . .	29
25.8.1 Detailed Description . . . . .	30
25.8.2 Field Documentation . . . . .	30
<b>26 File Documentation</b>	<b>32</b>
26.1 approximating.c File Reference . . . . .	32
26.1.1 Detailed Description . . . . .	33
26.1.2 Define Documentation . . . . .	33
26.1.3 Function Documentation . . . . .	34
26.2 compute.c File Reference . . . . .	36
26.2.1 Detailed Description . . . . .	36
26.2.2 Function Documentation . . . . .	36
26.3 distribution.c File Reference . . . . .	37
26.3.1 Detailed Description . . . . .	38
26.3.2 Define Documentation . . . . .	38
26.3.3 Function Documentation . . . . .	38
26.4 flipsign.c File Reference . . . . .	40

26.4.1 Detailed Description . . . . .	41
26.4.2 Define Documentation . . . . .	41
26.4.3 Function Documentation . . . . .	41
26.5 ksp.c File Reference . . . . .	42
26.5.1 Define Documentation . . . . .	44
26.5.2 Function Documentation . . . . .	44
26.5.3 Variable Documentation . . . . .	46
26.6 kspmonitor.c File Reference . . . . .	47
26.6.1 Define Documentation . . . . .	47
26.6.2 Function Documentation . . . . .	48
26.6.3 Variable Documentation . . . . .	48
26.7 linear.c File Reference . . . . .	49
26.7.1 Detailed Description . . . . .	50
26.7.2 Function Documentation . . . . .	50
26.8 linear_impl.h File Reference . . . . .	59
26.8.1 Define Documentation . . . . .	60
26.9 linksp.h File Reference . . . . .	62
26.9.1 Function Documentation . . . . .	62
26.10linpc.h File Reference . . . . .	62
26.10.1 Define Documentation . . . . .	63
26.10.2 Function Documentation . . . . .	65
26.11Make.inc File Reference . . . . .	66
26.12options.c File Reference . . . . .	66
26.12.1 Define Documentation . . . . .	66
26.12.2 Function Documentation . . . . .	67
26.13pc.c File Reference . . . . .	67
26.13.1 Define Documentation . . . . .	68
26.13.2 Function Documentation . . . . .	68
26.14pcstuff.c File Reference . . . . .	70
26.14.1 Function Documentation . . . . .	71
26.15preprocess.c File Reference . . . . .	72

26.15.1 Define Documentation . . . . .	75
26.15.2 Typedef Documentation . . . . .	75
26.15.3 Function Documentation . . . . .	75
26.15.4 Variable Documentation . . . . .	85
26.16reporting.c File Reference . . . . .	87
26.16.1 Define Documentation . . . . .	88
26.16.2 Function Documentation . . . . .	89
26.16.3 Variable Documentation . . . . .	93
26.17scaling.c File Reference . . . . .	94
26.17.1 Detailed Description . . . . .	94
26.17.2 Define Documentation . . . . .	95
26.17.3 Function Documentation . . . . .	95
26.18singleton.c File Reference . . . . .	96
26.18.1 Detailed Description . . . . .	97
26.18.2 Define Documentation . . . . .	97
26.18.3 Function Documentation . . . . .	98
26.19suit.c File Reference . . . . .	99
26.19.1 Function Documentation . . . . .	100
26.19.2 Suitability functions for the linear problem . . . . .	100
26.20syspro.h File Reference . . . . .	100
26.20.1 Define Documentation . . . . .	103
26.20.2 Typedef Documentation . . . . .	103
26.20.3 Function Documentation . . . . .	104
26.21syspro_anamod.c File Reference . . . . .	119
26.21.1 Function Documentation . . . . .	119
26.22syspro_impl.h File Reference . . . . .	121
26.22.1 Define Documentation . . . . .	121
26.23sysprolinear.h File Reference . . . . .	122
26.23.1 Typedef Documentation . . . . .	124
26.23.2 Function Documentation . . . . .	124
26.24sysprosuit.h File Reference . . . . .	135

26.24.1 Function Documentation . . . . .	136
26.24.2 Suitability functions for the linear problem . . . . .	136
26.25sysprotransform.h File Reference . . . . .	136
26.25.1 Function Documentation . . . . .	138
26.26testmat.c File Reference . . . . .	151
26.26.1 Function Documentation . . . . .	151
26.26.2 Variable Documentation . . . . .	153
26.27testmat16.c File Reference . . . . .	154
26.27.1 Function Documentation . . . . .	154
26.27.2 Variable Documentation . . . . .	155
26.28tracing.c File Reference . . . . .	155
26.28.1 Function Documentation . . . . .	155
26.28.2 Variable Documentation . . . . .	157
26.29transform.c File Reference . . . . .	158
26.29.1 Define Documentation . . . . .	160
26.29.2 Function Documentation . . . . .	160
26.30u1.c File Reference . . . . .	172
26.30.1 Function Documentation . . . . .	172
26.31u12.c File Reference . . . . .	172
26.31.1 Function Documentation . . . . .	173
26.32u13.c File Reference . . . . .	173
26.32.1 Function Documentation . . . . .	174
26.33u14.c File Reference . . . . .	175
26.33.1 Function Documentation . . . . .	176
26.34u15.c File Reference . . . . .	178
26.34.1 Function Documentation . . . . .	179
26.35u16.c File Reference . . . . .	180
26.35.1 Function Documentation . . . . .	181
26.36u2.c File Reference . . . . .	183
26.36.1 Function Documentation . . . . .	183
26.37u3.c File Reference . . . . .	184

<a href="#">26.37.1 Function Documentation</a> . . . . .	184
<a href="#">26.38u4.c File Reference</a> . . . . .	186
<a href="#">26.38.1 Function Documentation</a> . . . . .	187

## 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 possible 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 [Preprocessed-Solution\(\)](#) 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 possibly 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 predominantly 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 [compute.c](#).

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 preserve 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 for 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:

<a href="#">LinearSolution_</a>	12
<a href="#">LinearSystem_</a>	14
<a href="#">NumericalProblem_</a>	17
<a href="#">PreprocessorsGlobalInfo_</a>	18

<a href="#">SalsaTransform_</a>	21
<a href="#">SalsaTransformObject_</a>	24
<a href="#">singleton_struct</a>	28
<a href="#">SystemPreprocessor_</a>	29

## 24 File Index

### 24.1 File List

Here is a list of all files with brief descriptions:

<a href="#">approximating.c</a>	32
<a href="#">compute.c</a> (System/Anamod and NMD interface )	36
<a href="#">distribution.c</a>	37
<a href="#">flipsign.c</a>	40
<a href="#">ksp.c</a>	42
<a href="#">kspmonitor.c</a>	47
<a href="#">linear.c</a>	49
<a href="#">linear_impl.h</a>	59
<a href="#">linksp.h</a>	62
<a href="#">linpc.h</a>	62
<a href="#">Make.inc</a>	66
<a href="#">options.c</a>	66
<a href="#">pc.c</a>	67
<a href="#">pcstuff.c</a>	70
<a href="#">preprocess.c</a>	72
<a href="#">reporting.c</a>	87
<a href="#">scaling.c</a>	94

<a href="#">singleton.c</a>	96
<a href="#">suit.c</a>	99
<a href="#">syspro.h</a>	100
<a href="#">syspro_anamod.c</a>	119
<a href="#">syspro_impl.h</a>	121
<a href="#">sysprolinear.h</a>	122
<a href="#">sysprosuit.h</a>	135
<a href="#">sysprotransform.h</a>	136
<a href="#">testmat.c</a>	151
<a href="#">testmat16.c</a>	154
<a href="#">tracing.c</a>	155
<a href="#">transform.c</a>	158
<a href="#">u1.c</a>	172
<a href="#">u12.c</a>	172
<a href="#">u13.c</a>	173
<a href="#">u14.c</a>	175
<a href="#">u15.c</a>	178
<a href="#">u16.c</a>	180
<a href="#">u2.c</a>	183
<a href="#">u3.c</a>	184
<a href="#">u4.c</a>	186

## 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()`, `LinearSystemGetTempVector()`, `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(\* [metadataacomputer](#) )(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 `TransformReportTeXTTable()`.

#### 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 \*, [Suitability-Value](#) \*)
- 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(), TransformObjectGetName(), 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 TransformObjectAddOptionExplanation().

**25.6.2.13 int \* SalsaTransformObject\_::options**

Definition at line 35 of file syspro\_impl.h.

Referenced by FreeTransformObject(), TransformItemDescribeLong(), TransformItemGetFirstOption(), TransformItemGetNextOption(), TransformItemOptionMark(), 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(), TransformItemGetNextOption(), 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 TransformObjectSetSuitabilityFunction().

### 25.6.2.16 PetscErrorCode(\* SalsaTransformObject\_::suitabilityfunction)(NumericalProblem, void \*, SuitabilityValue \*)

Referenced by TransformObjectGetSuitabilityFunction(), and TransformObjectSetSuitabilityFunction().

### 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\_ksp().

### 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(), LinearSystemGetParts(), 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 flpsign.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 "flpsign"

### Functions

- static PetscErrorCode [flpsign](#) (char \*type, int nopt, PetscTruth overwrite, NumericalProblem inproblem, NumericalProblem \*outproblem, void \*gctx, void \*\*ctx, PetscTruth \*success)
- static PetscErrorCode [back\\_flpsign](#) (char \*flpsign\_type, PetscTruth overwrite, void \*gctx, void \*ctx, NumericalProblem nextproblem, NumericalProblem problem, NumericalSolution flipped, NumericalSolution straight)

- static PetscErrorCode [setup\\_flpsign\\_choices](#) ()
- static PetscErrorCode [specific\\_flpsign\\_choices](#) ([NumericalProblem](#) theproblem, [SalsaTransform](#) flpsign)
- PetscErrorCode [DeclareFlpsignPreprocessor](#) (void)

### 26.4.1 Detailed Description

Definition in file [flpsign.c](#).

### 26.4.2 Define Documentation

#### 26.4.2.1 #define PREPROCESSOR "flpsign"

Definition at line 19 of file flpsign.c.

### 26.4.3 Function Documentation

#### 26.4.3.1 static PetscErrorCode back\_flpsign (char \* *flpsign\_type*, PetscTruth *overwrite*, void \* *gctx*, void \* *ctx*, [NumericalProblem](#) *nextproblem*, [NumericalProblem](#) *problem*, [NumericalSolution](#) *flipped*, [NumericalSolution](#) *straight*) [static]

Definition at line 85 of file flpsign.c.

References [CHKERRQ\(\)](#), [DeleteLinearSystem\(\)](#), [ierr](#), and [LinearSolutionCopy\(\)](#).

Referenced by [DeclareFlpsignPreprocessor\(\)](#).

#### 26.4.3.2 PetscErrorCode DeclareFlpsignPreprocessor (void)

Definition at line 168 of file flpsign.c.

References [back\\_flpsign\(\)](#), [CHKERRQ\(\)](#), [DeclarePreprocessor\(\)](#), [ierr](#), [PREPROCESSOR](#), [PreprocessorSetPreservedCategories\(\)](#), [setup\\_flpsign\\_choices\(\)](#), and [specific\\_flpsign\\_choices\(\)](#).



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

Definition at line 24 of file flpsign.c.

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

**26.4.3.4** `static PetscErrorCode setup_flpsign_choices ()` `[static]`

This routine is only called when the flpsign preprocessor is created by `DeclarePreprocessor()` inside `DeclareFlpsignPreprocessor()`

Definition at line 114 of file flpsign.c.

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

Referenced by `DeclareFlpsignPreprocessor()`.

**26.4.3.5** `static PetscErrorCode specific_flpsign_choices (NumericalProblem theproblem, SalsaTransform flpsign)` `[static]`

This is the 'specific setup' phase of the flpsign 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 flpsign.c.

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

Referenced by `DeclareFlpsignPreprocessor()`.

## 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](#) (KSPTType kspt, PetscTruth \*f)
- static PetscErrorCode [setup\\_ksp\\_choices](#) ()
- static PetscErrorCode [unset\\_ksp](#) (NumericalProblem\_ \*dummy)
- static PetscErrorCode [disable\\_ksp](#) (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\_ksp(), ierr, PREPROCESSOR, PreprocessorSetPreservedCategories(), setup\_ksp(), setup\_ksp\_choices(), unset\_ksp(), and unset\_ksp().

### 26.5.2.2 static PetscErrorCode disable\_ksp (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\_ksp (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 SysProLinearInstallCustomKSPMonitor().

### 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 \*nrms)
- 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 *new*)

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 [SYSPROCHECKVALIDLINSOLa](#).

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 [SYSPROCHECKVALIDLINSOL](#).

#### 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 *new*)

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 SYSPROCHECKVALIDLINSYS.

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()`, `solverlinear()`, `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()`, `MatGustafssonMod()`, `MatSymmetricPart()`, and `setup_pc()`.

#### 26.7.2.32 PetscErrorCode LinearSystemTrueDistance (LinearSystem system, LinearSolution *linsol*, PetscReal \* *rnorm*)

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()`, `LinearSystemGetKnownSolution()`, `LinearSystemGetParts()`, `LinearSystemGetTmpVector()`, `LinearSystemTrueDistance()`, `SYSPROCHECKVALIDLINSOL`, and `SYSPROCHECKVALIDLINSYS`.

#### 26.7.2.34 PetscErrorCode PreprocessedLinearSystemSolution (LinearSystem *sys*, LinearSolution \* *sol*)

Definition at line 705 of file `linear.c`.

References `CHKERRQ()`, `ierr`, `PreprocessedProblemSolving()`, `RegisterPreprocessorContext()`, 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) {SYSPROCHECK-  
VALID(i,LINSYSCookie,"linear system");}
- #define [SYSPROCHECKVALIDLINSYSa](#)(i, a) {SYSPROCHECKVAL-  
IDa(i,LINSYSCookie,"linear system",a);}
- #define [SYSPROCHECKVALIDLINSOL](#)(i) {SYSPROCHECK-  
VALID(i,LINSOLCookie,"linear solution");}
- #define [SYSPROCHECKVALIDLINSOLa](#)(i, a) {SYSPROCHECKVAL-  
IDa(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 [LinearSystem-InheritParts\(\)](#).

#### 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 PCRAMS "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\(\)](#), [SetPetscOptionsForPC\(\)](#), 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(), TransformObjectGetByName(), 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 TransformObjectMark().

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 *),
    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, Numerical-
  Problem 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, Nu-
  mericalSolution *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 specifying 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_::classtaticsetup`, `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`, `SuccessorPre-`

processor(), SysProPreprocessorEndFunction(), SysproPreprocessorStartFunction(), SysProProblemCloneContext(), SysProTraceMessage(), SystemPreprocessorGetByName(), SystemPreprocessor\_::transform, TransformGetNextUnmarkedItem(), TransformGetUserChoices(), TransformItemGetFirstOption(), TransformItemGetNextOption(), 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 solutioncontext.

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 SystemPreprocessorGetByName().

### 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\_ksp().

### 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 `unsetpreprocessor`.

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`, `currentpreprocessors`, `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 SysProGetContextFunc-  
 tions().

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(), De-  
 declarePreprocessorRequiredCategories(), PreprocessedSolution(), PreprocessorGetPre-

servedCategories(), 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 \*flag)
- 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 specifying 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_::classtaticsetup`, `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, PreprocessorSaveAprioriSelection(), ReportEnabledPreprocessors(), SystemPreprocessorGetByName(), TransformGetByName(), TransformGetNextUnmarkedItem(), TransformGetNUnmarked(), TransformItemOptionsUseOnly(), TransformObjectGetName(), TransformObjectsUseOnly(), 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 solutioncontext.

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 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().

### 26.20.3.20 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().

#### 26.20.3.21 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 preprocess-readout.

**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 \* *flag*)**

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`, `preprocessor-contexts`, `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 *flag)`

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`, `current-preprocessors`, `ierr`, `NPREPROCESS`, `preprocessorcontexts`, and `unsetpreprocessor`.

Referenced by `main()`.

### 26.20.3.40 `PetscErrorCode SysProRemoveQuantity (NumericalProblem theproblem, char *cat, char *cmp, PetscTruth *flag)`

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 \****flag*)

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 \* *flag*)

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 \* *flag*)

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`, `PREPROCESSOR`, `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_ksp()`, `ierr`, `PREPROCESSOR`, `PreprocessorSetPreservedCategories()`, `setup_ksp()`, `setup_ksp_choices()`, `unset_ksp()`, and `unset_ksp()`.

#### 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 `SYSPROCHECKVALIDDLINSOL`.

### 26.23.2.18 PetscErrorCode LinearSolutionAddToPreprocessTime (LinearSolution, PetscLogDouble)

### 26.23.2.19 PetscErrorCode LinearSolutionCopy (LinearSolution *old*, LinearSolution *new*)

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 `SYSPROCHECKVALIDDLINSOLa`.

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 [SYSPROCHECKVALIDLINSOLa](#).

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 [SYSPROCHECKVALIDLINSOL](#).

### 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 *new***)

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 `SYSPROCHECKVALIDLINSYS`.

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()`, `solvilinear()`, `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(), MatGustafssonMod(), 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(), LinearSystemGetKnownSolution(), LinearSystemGetParts(), LinearSystemGetTmpVector(), LinearSystemTrueDistance(), SYSPROCHECKVALIDLINSOL, and SYSPROCHECKVALIDLINSYS.

**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(), RegisterPreprocessorContext(), 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, CHKERRQ(), 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, TransformGetName(), 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 an 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 \* snw, 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 PreprocessorOptionsHandling().

#### 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\_ksp(), 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 TransformObjectGetByName().

**26.25.1.31 PetscErrorCode TransformItemOptionsUseOnly (SalsaTransformObject, char \*)**

Definition at line 555 of file transform.c.

References CHKERRQ(), ierr, SalsaTransformObject\_::n\_options, and TransformObjectAddOption().

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 TransformItemOptionsUseOnly().



### 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\_ksp(), 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\_ksp().

### 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\_ksp(), 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\_ksp(), 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\_ksp().

#### **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\_ksp(), disable\_pcs(), pcoptionshandling(), PreprocessorSpecificSetup(), 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 \* *ctx*,**  
**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 `TransformReportEnabled()`.

**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(), DeclareKSPPPreprocessor(), DeclarePCPreprocessor(),

DeclarePreprocessor(), DeclarePreprocessorIntelligentChoice(), DeclarePreprocessorRequiredCategories(), DeclareScalingPreprocessor(), DeclareSingletonPreprocessor(), DeleteLinearSystem(), delintctx(), delprob(), delsol(), DeregisterTransform(), destroy\_solver(), destroysolution(), disable\_ksp(), disable\_pcs(), distribute\_system(), eliminate\_singletons(), flpsign(), 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\_flpsign\_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\_flpsign\_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\_ksp(), 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\_ksp(), 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\_ksp(), 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 \**flag*)

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(\*)([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, CHKERRQ(), 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, TransformGetByName(), 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 PreprocessorOptionsHandling().

**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\_ksp(), 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 \* *flag*)**

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 TransformObjectGetByName().

### 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 TransformObjectAddOption().

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\_ksp(), 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\_ksp().

**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\_ksp(), 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\_ksp(), 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\_ksp().

**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_ksp()`, `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, TransformObjectGetMark(), 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 \**new*, 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 detsol (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(), detsol(), 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

### A

- LinearSystem\_, 14
- active\_option
  - SalsaTransformObject\_, 25
- adder
  - u3.c, 184
  - u4.c, 187
- alloc\_annotate\_c
  - SalsaTransformObject\_, 25
- alloc\_annotate\_i
  - SalsaTransformObject\_, 25
- alloc\_objects
  - SalsaTransform\_, 22
- alloc\_options
  - SalsaTransformObject\_, 25
- ALLPARTSNEW
  - linear\_impl.h, 60
- annotate\_c
  - SalsaTransformObject\_, 25
- annotate\_i
  - SalsaTransformObject\_, 26
- annotations\_c
  - SalsaTransform\_, 22
- annotations\_i
  - SalsaTransform\_, 22
- approximate\_system
  - approximating.c, 34
- approximating.c, 32
  - approximate\_system, 34
  - DeclareApproximationPreprocessor, 34
  - MatGustafssonMod, 34
  - MatSymmetricPart, 34
  - PREPROCESSOR, 33
  - setup\_approximation\_choices, 35
  - specific\_approximation\_choices, 35
  - unapproximate\_system, 35
- aprioriselection
  - SalsaTransform\_, 22

### B

- LinearSystem\_, 14
- back\_flipsign

- flipsign.c, 41

- back\_singleton
  - singleton.c, 98

### CHKERRQ

- testmat.c, 151
- testmat16.c, 154

### ChooseFirstTransform

- preprocess.c, 75

### classdynamicsetup

- PreprocessorsGlobalInfo\_, 18

### classproblemcloner

- PreprocessorsGlobalInfo\_, 19

### classtaticsetup

- PreprocessorsGlobalInfo\_, 19

### clonecontext

- PreprocessorsGlobalInfo\_, 19

### comm

- LinearSystem\_, 15
- NumericalProblem\_, 17

### compute.c, 36

- DeclarePreprocessorRequiredCategories, 36

- PreprocessorGetPreservedCategories, 37

- PreprocessorSetPreservedCategories, 37

### computecategory

- PreprocessorsGlobalInfo\_, 19

### ContinueRetrievingAllPreprocessors

- reporting.c, 89

- syspro.h, 104

### ContinueRetrievingCurrentPreprocessors

- reporting.c, 89

- syspro.h, 104

### cookie

- LinearSolution\_, 13

- LinearSystem\_, 15

### copy

- u1.c, 172

### create\_solver

- pc.c, 68

- u13.c, 174

- u14.c, [176](#)
- u15.c, [179](#)
- u16.c, [181](#)
- CreateDefaultLinearSolution
  - linear.c, [50](#)
  - sysprolinear.h, [124](#)
- CreateGlobalInfo
  - preprocess.c, [76](#)
- CreateLinearSolution
  - linear.c, [50](#)
  - sysprolinear.h, [124](#)
- CreateLinearSystem
  - linear.c, [51](#)
  - sysprolinear.h, [125](#)
- ctx
  - LinearSolution\_, [13](#)
  - LinearSystem\_, [15](#)
  - NumericalProblem\_, [17](#)
- ctxcreate
  - SystemPreprocessor\_, [30](#)
- ctxdelete
  - SystemPreprocessor\_, [30](#)
- currentchoices
  - preprocess.c, [85](#)
- currentoptions
  - preprocess.c, [85](#)
- currentpreprocessors
  - preprocess.c, [85](#)
- custommonitor
  - ksp.c, [46](#)
- declareadders
  - u3.c, [184](#)
  - u4.c, [187](#)
- DeclareApproximationPreprocessor
  - approximating.c, [34](#)
  - sysprolinear.h, [125](#)
- DeclareDistributionPreprocessor
  - distribution.c, [38](#)
  - sysprolinear.h, [125](#)
- DeclareDummyRowPreprocessor
  - sysprolinear.h, [125](#)
- DeclareFlipsignPreprocessor
  - flipsign.c, [41](#)
  - sysprolinear.h, [125](#)
- DeclareKSPPreprocessor
  - ksp.c, [44](#)
  - sysprolinear.h, [126](#)
- DeclarePCPreprocessor
  - pc.c, [68](#)
  - sysprolinear.h, [126](#)
- DeclarePreprocessor
  - preprocess.c, [76](#)
  - syspro.h, [104](#)
- DeclarePreprocessorIntelligentChoice
  - preprocess.c, [77](#)
  - syspro.h, [106](#)
- DeclarePreprocessorRequiredCategories
  - compute.c, [36](#)
- DeclareScalingPreprocessor
  - scaling.c, [95](#)
  - sysprolinear.h, [126](#)
- DeclareSingletonPreprocessor
  - singleton.c, [98](#)
  - sysprolinear.h, [126](#)
- delete\_diagnostics
  - sysprolinear.h, [127](#)
- DeleteLinearSystem
  - linear.c, [51](#)
  - sysprolinear.h, [127](#)
- delintctx
  - u4.c, [187](#)
- delprob
  - u3.c, [185](#)
  - u4.c, [187](#)
- delsol
  - u3.c, [185](#)
  - u4.c, [187](#)
- DeregisterTransform
  - sysprotransform.h, [138](#)
  - transform.c, [160](#)
- destroy\_solver
  - pc.c, [69](#)
  - u13.c, [174](#)
  - u14.c, [176](#)
  - u15.c, [179](#)
  - u16.c, [181](#)
- destroysolution
  - u13.c, [174](#)
- Diagnostics
  - sysprolinear.h, [124](#)
- disable\_ksp

- ksp.c, [44](#)
- disable\_pcs
  - pc.c, [69](#)
- distribute\_system
  - distribution.c, [38](#)
- distribution.c, [37](#)
  - DeclareDistributionPreprocessor, [38](#)
  - distribute\_system, [38](#)
  - PREPROCESSOR, [38](#)
  - sans\_partition, [39](#)
  - setup\_distribution\_choices, [39](#)
  - specific\_distribution\_choices, [39](#)
  - SpectrumComputeUnpreconditionedSpectrum, [39](#)
  - undistribute\_system, [40](#)
- eliminate\_singletons
  - singleton.c, [98](#)
- end\_function
  - SystemPreprocessor\_, [30](#)
- errortracer
  - PreprocessorsGlobalInfo\_, [19](#)
- estimate\_completion\_from\_hist
  - kspmonitor.c, [48](#)
- exhaustive
  - SystemPreprocessor\_, [30](#)
- explanation
  - SalsaTransformObject\_, [26](#)
- extractor
  - singleton\_struct, [29](#)
- flipsign
  - flipsign.c, [41](#)
- flipsign.c, [40](#)
  - back\_flipsign, [41](#)
  - DeclareFlipsignPreprocessor, [41](#)
  - flipsign, [41](#)
  - PREPROCESSOR, [41](#)
  - setup\_flipsign\_choices, [42](#)
  - specific\_flipsign\_choices, [42](#)
- for
  - testmat.c, [152](#)
  - testmat16.c, [154](#)
- freecontext
  - PreprocessorsGlobalInfo\_, [20](#)
- FreeTransformObject
  - sysprotransform.h, [138](#)
  - transform.c, [160](#)
- get\_pc\_stats\_function
  - pcstuff.c, [71](#)
- GetFirstPreprocessor
  - reporting.c, [89](#)
  - syspro.h, [106](#)
- GetNextPreprocessor
  - reporting.c, [89](#)
  - syspro.h, [107](#)
- GlobalInfo
  - preprocess.c, [85](#)
- gmrescycleid
  - ksp.c, [46](#)
  - kspmonitor.c, [48](#)
- ierr
  - testmat.c, [153](#)
  - testmat16.c, [155](#)
- ilu\_stats\_function
  - pcstuff.c, [71](#)
- Init
  - LinearSystem\_, [15](#)
- InitRetrievingPreprocessors
  - reporting.c, [90](#)
  - syspro.h, [107](#)
- intelligence
  - SystemPreprocessor\_, [30](#)
- is\_gmres\_method
  - ksp.c, [44](#)
- ITER\_DIVERGENCE
  - kspmonitor.c, [47](#)
- ITER\_STAGNATION
  - kspmonitor.c, [48](#)
- known\_solution
  - LinearSystem\_, [15](#)
- ksp.c, [42](#)
  - custommonitor, [46](#)
  - DeclareKSPPreprocessor, [44](#)
  - disable\_ksps, [44](#)
  - gmrescycleid, [46](#)
  - is\_gmres\_method, [44](#)
  - monitordata, [47](#)
  - PREPROCESSOR, [44](#)

- set\_ksp\_options, [44](#)
- setup\_ksp, [45](#)
- setup\_ksp\_choices, [45](#)
- SysProLinearDeclareCustomKSP-Monitor, [45](#)
- SysProLinearInstallCustomKSP-Monitor, [45](#)
- unset\_ksp, [46](#)
- unset\_ksp, [46](#)
- kspmonitor.c, [47](#)
- estimate\_completion\_from\_hist, [48](#)
- gmrescycleid, [48](#)
- ITER\_DIVERGENCE, [47](#)
- ITER\_STAGNATION, [48](#)
- MonitorAdjustMaxit, [48](#)
- linear.c, [49](#)
  - CreateDefaultLinearSolution, [50](#)
  - CreateLinearSolution, [50](#)
  - CreateLinearSystem, [51](#)
  - DeleteLinearSystem, [51](#)
  - LinearCopyNumericalSolution, [51](#)
  - LinearCreateNumericalSolution, [52](#)
  - LinearDeleteNumericalSolution, [52](#)
  - LinearDeleteNumericalSolution-Context, [52](#)
  - LinearPackageSetUp, [52](#)
  - LinearSolutionCopy, [52](#)
  - LinearSolutionCopyStats, [53](#)
  - LinearSolutionCreateStatistics, [53](#)
  - LinearSolutionDelete, [53](#)
  - LinearSolutionGetContext, [54](#)
  - LinearSolutionGetStatistics, [54](#)
  - LinearSolutionGetVector, [54](#)
  - LinearSolutionSetContext, [54](#)
  - LinearSolutionSetVector, [54](#)
  - LinearSystemCopy, [55](#)
  - LinearSystemDuplicate, [55](#)
  - LinearSystemDuplicatePointers, [55](#)
  - LinearSystemGetContext, [56](#)
  - LinearSystemGetKnownSolution, [56](#)
  - LinearSystemGetMetadata, [56](#)
  - LinearSystemGetParts, [57](#)
  - LinearSystemGetTmpVector, [57](#)
  - LinearSystemInheritParts, [57](#)
  - LinearSystemSetContext, [57](#)
  - LinearSystemSetKnownSolution, [58](#)
  - LinearSystemSetMetadata, [58](#)
  - LinearSystemSetParts, [58](#)
  - LinearSystemTrueDistance, [59](#)
  - LinearSystemTrueDistancePrint, [59](#)
  - PreprocessedLinearSystemSolution, [59](#)
- linear\_impl.h, [59](#)
  - ALLPARTSNEW, [60](#)
  - LINSOLCOOKIE, [60](#)
  - LINSYSCookie, [60](#)
  - SYSPROCHECKVALIDLINSOL, [61](#)
  - SYSPROCHECKVALIDLINSOLa, [61](#)
  - SYSPROCHECKVALIDLINSYS, [61](#)
  - SYSPROCHECKVALIDLINSYSa, [61](#)
- LinearCopyNumericalSolution
  - linear.c, [51](#)
  - sysprolinear.h, [127](#)
- LinearCreateNumericalSolution
  - linear.c, [52](#)
  - sysprolinear.h, [127](#)
- LinearDeleteNumericalSolution
  - linear.c, [52](#)
  - sysprolinear.h, [127](#)
- LinearDeleteNumericalSolutionContext
  - linear.c, [52](#)
  - sysprolinear.h, [128](#)
- LinearPackageSetUp
  - linear.c, [52](#)
- LinearSolution
  - sysprolinear.h, [124](#)
- LinearSolution\_, [12](#)
  - cookie, [13](#)
  - ctx, [13](#)
  - Out, [13](#)
  - statistics, [13](#)
- LinearSolutionAddToPreprocessTime
  - sysprolinear.h, [128](#)
- LinearSolutionCopy
  - linear.c, [52](#)
  - sysprolinear.h, [128](#)



- LinearSolutionCopyStats
  - linear.c, 53
  - sysprolinear.h, 128
- LinearSolutionCreateStatistics
  - linear.c, 53
  - sysprolinear.h, 129
- LinearSolutionDelete
  - linear.c, 53
  - sysprolinear.h, 129
- LinearSolutionGetContext
  - linear.c, 54
  - sysprolinear.h, 129
- LinearSolutionGetStatistics
  - linear.c, 54
  - sysprolinear.h, 129
- LinearSolutionGetTimes
  - sysprolinear.h, 130
- LinearSolutionGetVector
  - linear.c, 54
  - sysprolinear.h, 130
- LinearSolutionSetContext
  - linear.c, 54
  - sysprolinear.h, 130
- LinearSolutionSetTimes
  - sysprolinear.h, 130
- LinearSolutionSetVector
  - linear.c, 54
  - sysprolinear.h, 130
- LinearSystem
  - sysprolinear.h, 124
- LinearSystem\_
  - A, 14
  - B, 14
  - comm, 15
  - cookie, 15
  - ctx, 15
  - Init, 15
  - known\_solution, 15
  - metadata, 16
  - partsoriginal, 16
  - Rhs, 16
  - Sol, 16
  - Tmp, 16
- LinearSystemCopy
  - linear.c, 55
  - sysprolinear.h, 131
- LinearSystemDuplicate
  - linear.c, 55
  - sysprolinear.h, 131
- LinearSystemDuplicatePointers
  - linear.c, 55
  - sysprolinear.h, 131
- LinearSystemGetContext
  - linear.c, 56
  - sysprolinear.h, 132
- LinearSystemGetKnownSolution
  - linear.c, 56
  - sysprolinear.h, 132
- LinearSystemGetMetadata
  - linear.c, 56
  - sysprolinear.h, 132
- LinearSystemGetParts
  - linear.c, 57
  - sysprolinear.h, 132
- LinearSystemGetTmpVector
  - linear.c, 57
  - sysprolinear.h, 133
- LinearSystemInheritParts
  - linear.c, 57
  - sysprolinear.h, 133
- LinearSystemSetContext
  - linear.c, 57
  - sysprolinear.h, 133
- LinearSystemSetKnownSolution
  - linear.c, 58
  - sysprolinear.h, 133
- LinearSystemSetMetadata
  - linear.c, 58
  - sysprolinear.h, 134
- LinearSystemSetParts
  - linear.c, 58
  - sysprolinear.h, 134
- LinearSystemTrueDistance
  - linear.c, 59
  - sysprolinear.h, 134
- LinearSystemTrueDistancePrint
  - linear.c, 59
  - sysprolinear.h, 135
- LINLEN
  - reporting.c, 88
- linksp.h, 62

- SysProLinearDeclareCustomKSP-Monitor, 62
- SysProLinearInstallCustomKSP-Monitor, 62
- linpc.h, 62
  - pc\_short\_string, 65
  - PCBOOMERAMG, 63
  - PCBS95, 63
  - PCEUCLID, 63
  - PCMUMPS, 64
  - PCPARASAILS, 64
  - PCPILUT, 64
  - PCRASM, 64
  - PCSILU, 64
  - PCSPOOLES, 64
  - PCSUPERLU, 65
  - PCUMFPACK, 65
  - set\_pc\_options, 65
  - set\_preconditioner\_base\_matrix, 65
  - SetPetscOptionsForPC, 66
- LINSOLCOOKIE
  - linear\_impl.h, 60
- LINSYSCookie
  - linear\_impl.h, 60
- main
  - u1.c, 172
  - u12.c, 173
  - u13.c, 174
  - u14.c, 176
  - u15.c, 179
  - u16.c, 181
  - u2.c, 183
  - u3.c, 185
  - u4.c, 188
- Make.inc, 66
- make\_diagnostics
  - sysprolinear.h, 135
- makeintctx
  - u4.c, 188
- makesol
  - u3.c, 185
  - u4.c, 188
- marked
  - SalsaTransformObject\_, 26
- MatGustafssonMod
  - approximating.c, 34
- MatSymmetricPart
  - approximating.c, 34
- MAXLEN
  - reporting.c, 88
- metadata
  - LinearSystem\_, 16
- metadatacomputer
  - PreprocessorsGlobalInfo\_, 20
- MonitorAdjustMaxit
  - kspmonitor.c, 48
- monitordata
  - ksp.c, 47
- n
  - singleton\_struct, 29
- n\_annotate\_c
  - SalsaTransform\_, 23
- n\_annotate\_i
  - SalsaTransform\_, 23
- n\_objects
  - SalsaTransform\_, 23
- n\_options
  - SalsaTransformObject\_, 26
- name
  - SalsaTransform\_, 23
  - SalsaTransformObject\_, 26
  - SystemPreprocessor\_, 31
- NewTransform
  - sysprotransform.h, 138
  - transform.c, 160
- NewTransformObject
  - sysprotransform.h, 139
  - transform.c, 161
- NPREPROCESS
  - preprocess.c, 75
- np preprocess
  - preprocess.c, 86
  - reporting.c, 93
- NumericalProblem
  - syspro.h, 103
- NumericalProblem\_
  - comm, 17
  - ctx, 17
- NumericalProblemGetComm
  - preprocess.c, 78

- syspro.h, [107](#)
- NUMERICALPROBLEMHEADER
  - syspro\_impl.h, [121](#)
- NumericalSolution
  - syspro.h, [103](#)
- onlyforsymmetricproblem
  - suit.c, [100](#)
  - sysprosuit.h, [136](#)
- option
  - SalsaTransformObject\_, [27](#)
- optionexplanation
  - SalsaTransformObject\_, [27](#)
- options
  - SalsaTransformObject\_, [27](#)
- options.c, [66](#)
  - PreprocessorsOptionsHandling, [67](#)
  - TYPELEN, [66](#)
- options\_marked
  - SalsaTransformObject\_, [27](#)
- optionshandling
  - SystemPreprocessor\_, [31](#)
- Out
  - LinearSolution\_, [13](#)
- partsoriginal
  - LinearSystem\_, [16](#)
- pc.c, [67](#)
  - create\_solver, [68](#)
  - DeclarePCPreprocessor, [68](#)
  - destroy\_solver, [69](#)
  - disable\_pcs, [69](#)
  - pcoptionshandling, [69](#)
  - PREPROCESSOR, [68](#)
  - setup\_pc, [69](#)
  - setup\_pc\_choices, [70](#)
  - unset\_pc, [70](#)
- pc\_short\_string
  - linpc.h, [65](#)
  - pcstuff.c, [71](#)
- pc\_string
  - pcstuff.c, [71](#)
- PCBOOMERAMG
  - linpc.h, [63](#)
- PCBS95
  - linpc.h, [63](#)
- PCEUCLID
  - linpc.h, [63](#)
- PCMUMPS
  - linpc.h, [64](#)
- pcoptionshandling
  - pc.c, [69](#)
- PCPARASAILS
  - linpc.h, [64](#)
- PCPILUT
  - linpc.h, [64](#)
- PCRASM
  - linpc.h, [64](#)
- PCSILU
  - linpc.h, [64](#)
- PCSPOOLES
  - linpc.h, [64](#)
- pcstuff.c, [70](#)
  - get\_pc\_stats\_function, [71](#)
  - ilu\_stats\_function, [71](#)
  - pc\_short\_string, [71](#)
  - pc\_string, [71](#)
  - set\_blocked\_sub\_pc, [71](#)
  - set\_preconditioner\_base\_matrix, [72](#)
  - SetPetscOptionsForPC, [72](#)
- PCSUPERLU
  - linpc.h, [65](#)
- PCUMFPACK
  - linpc.h, [65](#)
- preprocess.c, [72](#)
  - ChooseFirstTransform, [75](#)
  - CreateGlobalInfo, [76](#)
  - currentchoices, [85](#)
  - currentoptions, [85](#)
  - currentpreprocessors, [85](#)
  - DeclarePreprocessor, [76](#)
  - DeclarePreprocessorIntelligentChoice, [77](#)
  - GlobalInfo, [85](#)
  - NPREPROCESS, [75](#)
  - npreprocess, [86](#)
  - NumericalProblemGetComm, [78](#)
  - PreprocessedProblemSolving, [78](#)
  - PreprocessedSolution, [78](#)
  - preprocesslevel, [86](#)
  - preprocessorcontexts, [86](#)
  - PreprocessorGetContext, [79](#)

- PreprocessorGetIndex, 79
- PreprocessorGetSetting, 79
- preprocessors, 86
- PreprocessorsGlobalInfo, 75
- PreprocessorSpecificSetup, 79
- RegisterPreprocessorContext, 80
- RegisterPreprocessorSetting, 80
- RetrievePreprocessorChoice, 80
- solutioncontext, 86
- SysProDeclareErrorTracer, 81
- SysProDeclareFunctions, 81
- SysProDeclareProblemMonitor, 82
- SysProFinalize, 82
- SysProGetContextFunctions, 82
- SysProGetErrorTracer, 83
- SysProInitialize, 83
- SysProPreprocessorEndFunction, 83
- SysProPreprocessorStartFunction, 83
- SysProProblemCloneContext, 84
- SysProProblemDeleteContext, 84
- SystemPreprocessorGetByName, 84
- TransformGetByName, 85
- unsetpreprocessor, 87
- PreprocessedLinearSystemSolution
  - linear.c, 59
  - sysprolinear.h, 135
- PreprocessedProblemSolving
  - preprocess.c, 78
  - syspro.h, 107
- PreprocessedSolution
  - preprocess.c, 78
  - syspro.h, 108
- preprocesslevel
  - preprocess.c, 86
  - reporting.c, 93
- PREPROCESSOR
  - approximating.c, 33
  - distribution.c, 38
  - flipsign.c, 41
  - ksp.c, 44
  - pc.c, 68
  - scaling.c, 95
  - singleton.c, 97
- PreprocessorApplyAprioriSelection
  - sysprotransform.h, 139
- transform.c, 161
- preprocessorcontexts
  - preprocess.c, 86
- PreprocessorGetContext
  - preprocess.c, 79
  - syspro.h, 108
- PreprocessorGetIndex
  - preprocess.c, 79
  - syspro.h, 108
- PreprocessorGetPreservedCategories
  - compute.c, 37
  - syspro.h, 109
- PreprocessorGetSetting
  - preprocess.c, 79
  - syspro.h, 109
- preprocessors
  - preprocess.c, 86
  - reporting.c, 93
- PreprocessorSaveAprioriSelection
  - sysprotransform.h, 139
  - transform.c, 161
- PreprocessorSetPreservedCategories
  - compute.c, 37
  - syspro.h, 109
- PreprocessorsGlobalInfo
  - preprocess.c, 75
- PreprocessorsGlobalInfo\_, 18
  - classdynamicsetup, 18
  - classproblemcloner, 19
  - classstaticsetup, 19
  - clonecontext, 19
  - computecategory, 19
  - errortracer, 19
  - freecontext, 20
  - metadataacomputer, 20
  - problemdelete, 20
  - problemmonitor, 20
  - problemsolver, 20
  - solutioncontextdelete, 20
  - solutioncopy, 21
  - solutioncreator, 21
  - solutiondelete, 21
- PreprocessorsOptionsHandling
  - options.c, 67
  - syspro.h, 109
- PreprocessorSpecificSetup

- preprocess.c, 79
- preprocessreadout
  - reporting.c, 93
- preserved
  - SystemPreprocessor\_, 31
- problemdelete
  - PreprocessorsGlobalInfo\_, 20
- problemmonitor
  - PreprocessorsGlobalInfo\_, 20
- problemsolver
  - PreprocessorsGlobalInfo\_, 20
- ProcessPreprocessorOptions
  - syspro.h, 110
- RegisterPreprocessorContext
  - preprocess.c, 80
  - syspro.h, 110
- RegisterPreprocessorSetting
  - preprocess.c, 80
- ReportEnabledPreprocessors
  - reporting.c, 90
  - syspro.h, 110
- reporting.c, 87
  - ContinueRetrievingAllPreprocessors, 89
  - ContinueRetrievingCurrentPreprocessors, 89
  - GetFirstPreprocessor, 89
  - GetNextPreprocessor, 89
  - InitRetrievingPreprocessors, 90
  - LINELEN, 88
  - MAXLEN, 88
  - npprocess, 93
  - preprocesslevel, 93
  - preprocessors, 93
  - preprocessreadout, 93
  - ReportEnabledPreprocessors, 90
  - ReportSysProCallStackState, 90
  - REPOSITION, 88
  - RetrieveAllPreprocessorValues, 91
  - ScreenOutputTab, 91
  - ScreenOutputTabLine, 91
  - StartRetrievingAllPreprocessors, 91
  - StartRetrievingCurrentPreprocessors, 91
  - SuccessorPreprocessor, 92
  - TabReportActivePreprocessors, 92
  - TabReportAllPreprocessors, 92
  - TabReportPreprocessors, 92
- ReportSysProCallStackState
  - reporting.c, 90
  - syspro.h, 110
- REPOSITION
  - reporting.c, 88
- required
  - SystemPreprocessor\_, 31
- RetrieveAllPreprocessorValues
  - reporting.c, 91
  - syspro.h, 111
- RetrievePreprocessorChoice
  - preprocess.c, 80
  - syspro.h, 111
- Rhs
  - LinearSystem\_, 16
- SalsaTransform
  - syspro.h, 103
- SalsaTransform\_, 21
  - alloc\_objects, 22
  - annotations\_c, 22
  - annotations\_i, 22
  - aprioriselection, 22
  - n\_annotate\_c, 23
  - n\_annotate\_i, 23
  - n\_objects, 23
  - name, 23
  - transformobjects, 23
  - userchoices, 24
- SalsaTransformObject
  - syspro.h, 103
- SalsaTransformObject\_, 24
  - active\_option, 25
  - alloc\_annotate\_c, 25
  - alloc\_annotate\_i, 25
  - alloc\_options, 25
  - annotate\_c, 25
  - annotate\_i, 26
  - explanation, 26
  - marked, 26
  - n\_options, 26
  - name, 26
  - option, 27

- optionexplanation, 27
- options, 27
- options\_marked, 27
- suitabilityctx, 27
- suitabilityfunction, 28
- transform, 28
- sans\_partition
  - distribution.c, 39
- scale\_system
  - scaling.c, 95
- scaling.c, 94
  - DeclareScalingPreprocessor, 95
  - PREPROCESSOR, 95
  - scale\_system, 95
  - set\_intelligent\_scaling, 95
  - setup\_scaling\_choices, 95
  - specific\_scaling\_choices, 96
  - unscale\_system, 96
- ScreenOutputTab
  - reporting.c, 91
  - syspro.h, 111
- ScreenOutputTabLine
  - reporting.c, 91
  - syspro.h, 111
- set\_blocked\_sub\_pc
  - pcstuff.c, 71
- set\_intelligent\_scaling
  - scaling.c, 95
- set\_ksp\_options
  - ksp.c, 44
- set\_pc\_options
  - linpc.h, 65
- set\_preconditioner\_base\_matrix
  - linpc.h, 65
  - pcstuff.c, 72
- SetPetscOptionsForPC
  - linpc.h, 66
  - pcstuff.c, 72
- setup
  - SystemPreprocessor\_, 31
- setup\_approximation\_choices
  - approximating.c, 35
- setup\_distribution\_choices
  - distribution.c, 39
- setup\_flipsign\_choices
  - flipsign.c, 42
- setup\_ksp
  - ksp.c, 45
- setup\_ksp\_choices
  - ksp.c, 45
- setup\_pc
  - pc.c, 69
  - u13.c, 175
  - u14.c, 177
  - u15.c, 179
  - u16.c, 182
- setup\_pc\_choices
  - pc.c, 70
  - u13.c, 175
  - u14.c, 177
  - u15.c, 180
  - u16.c, 182
- setup\_scaling\_choices
  - scaling.c, 95
- setup\_singleton\_choices
  - singleton.c, 98
- singleton.c, 96
  - back\_singleton, 98
  - DeclareSingletonPreprocessor, 98
  - eliminate\_singletons, 98
  - PREPROCESSOR, 97
  - setup\_singleton\_choices, 98
  - singleton\_specific\_unset, 99
  - specific\_singleton\_choices, 99
- singleton\_specific\_unset
  - singleton.c, 99
- singleton\_struct, 28
  - extractor, 29
  - n, 29
  - t, 29
- Sol
  - LinearSystem\_, 16
- solutioncontext
  - preprocess.c, 86
- solutioncontextdelete
  - PreprocessorsGlobalInfo\_, 20
- solutioncopy
  - PreprocessorsGlobalInfo\_, 21
- solutioncreator
  - PreprocessorsGlobalInfo\_, 21
- solutiondelete
  - PreprocessorsGlobalInfo\_, 21

- solvebycopy
  - u2.c, [183](#)
  - u3.c, [185](#)
  - u4.c, [188](#)
- solvelinear
  - u12.c, [173](#)
  - u13.c, [175](#)
  - u14.c, [177](#)
  - u15.c, [180](#)
  - u16.c, [182](#)
- specific\_approximation\_choices
  - approximating.c, [35](#)
- specific\_distribution\_choices
  - distribution.c, [39](#)
- specific\_flipsign\_choices
  - flipsign.c, [42](#)
- specific\_scaling\_choices
  - scaling.c, [96](#)
- specific\_singleton\_choices
  - singleton.c, [99](#)
- SpectrumComputeUnpreconditionedSpectrum
  - distribution.c, [39](#)
- start\_function
  - SystemPreprocessor\_, [32](#)
- StartRetrievingAllPreprocessors
  - reporting.c, [91](#)
  - syspro.h, [111](#)
- StartRetrievingCurrentPreprocessors
  - reporting.c, [91](#)
  - syspro.h, [112](#)
- statistics
  - LinearSolution\_, [13](#)
- STRDUP
  - transform.c, [160](#)
- SuccessorPreprocessor
  - reporting.c, [92](#)
  - syspro.h, [112](#)
- suit.c, [99](#)
  - onlyforsymmetricproblem, [100](#)
- suitabilityctx
  - SalsaTransformObject\_, [27](#)
- suitabilityfunction
  - SalsaTransformObject\_, [28](#)
- SuitabilityValue
  - syspro.h, [103](#)
- syspro.h, [100](#)
  - ContinueRetrievingAllPreprocessors, [104](#)
  - ContinueRetrievingCurrentPreprocessors, [104](#)
  - DeclarePreprocessor, [104](#)
  - DeclarePreprocessorIntelligentChoice, [106](#)
  - GetFirstPreprocessor, [106](#)
  - GetNextPreprocessor, [107](#)
  - InitRetrievingPreprocessors, [107](#)
  - NumericalProblem, [103](#)
  - NumericalProblemGetComm, [107](#)
  - NumericalSolution, [103](#)
  - PreprocessedProblemSolving, [107](#)
  - PreprocessedSolution, [108](#)
  - PreprocessorGetContext, [108](#)
  - PreprocessorGetIndex, [108](#)
  - PreprocessorGetPreservedCategories, [109](#)
  - PreprocessorGetSetting, [109](#)
  - PreprocessorSetPreservedCategories, [109](#)
  - PreprocessorsOptionsHandling, [109](#)
  - ProcessPreprocessorOptions, [110](#)
  - RegisterPreprocessorContext, [110](#)
  - ReportEnabledPreprocessors, [110](#)
  - ReportSysProCallStackState, [110](#)
  - RetrieveAllPreprocessorValues, [111](#)
  - RetrievePreprocessorChoice, [111](#)
  - SalsaTransform, [103](#)
  - SalsaTransformObject, [103](#)
  - ScreenOutputTab, [111](#)
  - ScreenOutputTabLine, [111](#)
  - StartRetrievingAllPreprocessors, [111](#)
  - StartRetrievingCurrentPreprocessors, [112](#)
  - SuccessorPreprocessor, [112](#)
  - SuitabilityValue, [103](#)
  - SysProComputeQuantity, [112](#)
  - SysProDeclareErrorTracer, [113](#)
  - SysProDeclareFunctions, [113](#)
  - SysProDeclareProblemMonitor, [115](#)
  - SysProDeclareTraceContext, [115](#)
  - SysProDeclareTraceFunction, [115](#)
  - SysProDefaultTrace, [116](#)

- SysProFinalize, 116
- SysProFreeQuantities, 116
- SysProGetErrorTracer, 116
- SysProHasTrace, 117
- SysProInitialize, 117
- SysProRemoveQuantity, 117
- SysProRetrieveQuantity, 117
- SysProTraceMessage, 118
- SystemPreprocessor, 103
- SystemPreprocessorGetByName, 118
- TabReportActivePreprocessors, 118
- TabReportAllPreprocessors, 119
- TRUTH, 103
- syspro\_anamod.c, 119
  - SysProComputeQuantity, 119
  - SysProFreeQuantities, 120
  - SysProRemoveQuantity, 120
  - SysProRetrieveQuantity, 120
- syspro\_impl.h, 121
  - NUMERICALPROBLEM-HEADER, 121
  - SYSPROCHECKVALID, 121
  - SYSPROCHECKVALIDa, 122
- SYSPROCHECKVALID
  - syspro\_impl.h, 121
- SYSPROCHECKVALIDa
  - syspro\_impl.h, 122
- SYSPROCHECKVALIDLINSOL
  - linear\_impl.h, 61
- SYSPROCHECKVALIDLINSOLa
  - linear\_impl.h, 61
- SYSPROCHECKVALIDLINSYS
  - linear\_impl.h, 61
- SYSPROCHECKVALIDLINSYSa
  - linear\_impl.h, 61
- SysProComputeQuantity
  - syspro.h, 112
  - syspro\_anamod.c, 119
- SysProDeclareErrorTracer
  - preprocess.c, 81
  - syspro.h, 113
- SysProDeclareFunctions
  - preprocess.c, 81
  - syspro.h, 113
- SysProDeclareProblemMonitor
  - preprocess.c, 82
  - syspro.h, 115
- SysProDeclareTraceContext
  - syspro.h, 115
  - tracing.c, 155
- SysProDeclareTraceFunction
  - syspro.h, 115
  - tracing.c, 155
- SysProDefaultTrace
  - syspro.h, 116
  - tracing.c, 156
- SysProDefineCharAnnotation
  - sysprotransform.h, 140
  - transform.c, 162
- SysProDefineIntAnnotation
  - sysprotransform.h, 140
  - transform.c, 162
- SysProFinalize
  - preprocess.c, 82
  - syspro.h, 116
- SysProFreeQuantities
  - syspro.h, 116
  - syspro\_anamod.c, 120
- SysProGetContextFunctions
  - preprocess.c, 82
- SysProGetErrorTracer
  - preprocess.c, 83
  - syspro.h, 116
- SysProHasTrace
  - syspro.h, 117
  - tracing.c, 156
- SysProInitialize
  - preprocess.c, 83
  - syspro.h, 117
- sysprolinear.h, 122
  - CreateDefaultLinearSolution, 124
  - CreateLinearSolution, 124
  - CreateLinearSystem, 125
  - DeclareApproximationPreprocessor, 125
  - DeclareDistributionPreprocessor, 125
  - DeclareDummyRowPreprocessor, 125
  - DeclareFlipsignPreprocessor, 125
  - DeclareKSPPreprocessor, 126



- DeclarePCPreprocessor, 126
- DeclareScalingPreprocessor, 126
- DeclareSingletonPreprocessor, 126
- delete\_diagnostics, 127
- DeleteLinearSystem, 127
- Diagnostics, 124
- LinearCopyNumericalSolution, 127
- LinearCreateNumericalSolution, 127
- LinearDeleteNumericalSolution, 127
- LinearDeleteNumericalSolution-Context, 128
- LinearSolution, 124
- LinearSolutionAddToPreprocessTime, 128
- LinearSolutionCopy, 128
- LinearSolutionCopyStats, 128
- LinearSolutionCreateStatistics, 129
- LinearSolutionDelete, 129
- LinearSolutionGetContext, 129
- LinearSolutionGetStatistics, 129
- LinearSolutionGetTimes, 130
- LinearSolutionGetVector, 130
- LinearSolutionSetContext, 130
- LinearSolutionSetTimes, 130
- LinearSolutionSetVector, 130
- LinearSystem, 124
- LinearSystemCopy, 131
- LinearSystemDuplicate, 131
- LinearSystemDuplicatePointers, 131
- LinearSystemGetContext, 132
- LinearSystemGetKnownSolution, 132
- LinearSystemGetMetadata, 132
- LinearSystemGetParts, 132
- LinearSystemGetTmpVector, 133
- LinearSystemInheritParts, 133
- LinearSystemSetContext, 133
- LinearSystemSetKnownSolution, 133
- LinearSystemSetMetadata, 134
- LinearSystemSetParts, 134
- LinearSystemTrueDistance, 134
- LinearSystemTrueDistancePrint, 135
- make\_diagnostics, 135
- PreprocessedLinearSystemSolution, 135
- SysProLinearDeclareCustomKSPMonitor ksp.c, 45  
linksp.h, 62
- SysProLinearInstallCustomKSPMonitor ksp.c, 45  
linksp.h, 62
- SysProPreprocessorEndFunction preprocess.c, 83
- SysproPreprocessorStartFunction preprocess.c, 83
- SysProProblemCloneContext preprocess.c, 84
- SysProProblemDeleteContext preprocess.c, 84
- SysProRemoveQuantity syspro.h, 117  
syspro\_anamod.c, 120
- SysProRetrieveQuantity syspro.h, 117  
syspro\_anamod.c, 120
- sysprosuit.h, 135
- onlyforsymmetricproblem, 136
- sysprotrace tracing.c, 157
- sysprotracectx tracing.c, 157
- SysProTraceMessage syspro.h, 118  
tracing.c, 157
- sysprotransform.h, 136
- DeregisterTransform, 138
- FreeTransformObject, 138
- NewTransform, 138
- NewTransformObject, 139
- PreprocessorApplyAprioriSelection, 139
- PreprocessorSaveAprioriSelection, 139
- SysProDefineCharAnnotation, 140
- SysProDefineIntAnnotation, 140
- TransformCharAnnotationGetIndex, 140

- TransformCurrentItemDefineOption, 141
- TransformGetByName, 141
- TransformGetName, 141
- TransformGetNextUnmarkedItem, 141
- TransformGetNItems, 142
- TransformGetNUnmarked, 142
- TransformGetObjects, 142
- TransformGetUserChoices, 142
- TransformIntAnnotationGetIndex, 142
- TransformItemCharAnnotationGetIndex, 143
- TransformItemDefineOption, 143
- TransformItemDescribeLong, 143
- TransformItemDescribeShort, 143
- TransformItemGetCharAnnotation, 143
- TransformItemGetFirstOption, 144
- TransformItemGetIntAnnotation, 144
- TransformItemGetNextOption, 144
- TransformItemGetNOptions, 144
- TransformItemGetOptionI, 144
- TransformItemIntAnnotate, 144
- TransformItemOptionMark, 145
- TransformItemOptionsUseOnly, 145
- TransformObjectAddOption, 145
- TransformObjectAddOptionExplanation, 145
- TransformObjectCharAnnotate, 146
- TransformObjectDefineOption, 146
- TransformObjectGetByName, 146
- TransformObjectGetIntAnnotation, 146
- TransformObjectGetMark, 147
- TransformObjectGetName, 147
- TransformObjectGetSuitabilityFunction, 147
- TransformObjectGetTransformName, 148
- TransformObjectIntAnnotate, 148
- TransformObjectMark, 148
- TransformObjectSetExplanation, 148
- TransformObjectSetSuitabilityFunction, 149
- TransformObjectsGetNames, 149
- TransformObjectsMarkAll, 149
- TransformObjectsUnmarkAll, 149
- TransformObjectsUseOnly, 150
- TransformObjectUnmark, 150
- TransformReportEnabled, 150
- TransformReportTeXTable, 150
- TransformSetUserChoices, 151
- SystemPreprocessor
  - syspro.h, 103
- SystemPreprocessor\_, 29
  - ctxcreate, 30
  - ctxdelete, 30
  - end\_function, 30
  - exhaustive, 30
  - intelligence, 30
  - name, 31
  - optionshandling, 31
  - preserved, 31
  - required, 31
  - setup, 31
  - start\_function, 32
  - transform, 32
  - unset, 32
- SystemPreprocessorGetByName
  - preprocess.c, 84
  - syspro.h, 118
- t
  - singleton\_struct, 29
- TabReportActivePreprocessors
  - reporting.c, 92
  - syspro.h, 118
- TabReportAllPreprocessors
  - reporting.c, 92
  - syspro.h, 119
- TabReportPreprocessors
  - reporting.c, 92
- testmat.c, 151
  - CHKERRQ, 151
  - for, 152
  - ierr, 153
- testmat16.c, 154
  - CHKERRQ, 154

- for, [154](#)
- ierr, [155](#)
- TFINC
  - transform.c, [160](#)
- Tmp
  - LinearSystem\_, [16](#)
- tracing.c, [155](#)
  - SysProDeclareTraceContext, [155](#)
  - SysProDeclareTraceFunction, [155](#)
  - SysProDefaultTrace, [156](#)
  - SysProHasTrace, [156](#)
  - sysprotrace, [157](#)
  - sysprotracectx, [157](#)
  - SysProTraceMessage, [157](#)
- transform
  - SalsaTransformObject\_, [28](#)
  - SystemPreprocessor\_, [32](#)
- transform.c, [158](#)
  - DeregisterTransform, [160](#)
  - FreeTransformObject, [160](#)
  - NewTransform, [160](#)
  - NewTransformObject, [161](#)
  - PreprocessorApplyAprioriSelection, [161](#)
  - PreprocessorSaveAprioriSelection, [161](#)
  - STRDUP, [160](#)
  - SysProDefineCharAnnotation, [162](#)
  - SysProDefineIntAnnotation, [162](#)
  - TFINC, [160](#)
  - TransformCharAnnotationGetIndex, [162](#)
  - TransformGetName, [162](#)
  - TransformGetNextUnmarkedItem, [163](#)
  - TransformGetNUnmarked, [163](#)
  - TransformGetObjects, [163](#)
  - TransformGetUserChoices, [163](#)
  - TransformIntAnnotationGetIndex, [164](#)
  - TransformItemDescribeLong, [164](#)
  - TransformItemDescribeShort, [164](#)
  - TransformItemGetFirstOption, [164](#)
  - TransformItemGetNextOption, [165](#)
  - TransformItemOptionMark, [165](#)
  - TransformItemOptionsUseOnly, [165](#)
  - TransformObjectAddOption, [165](#)
  - TransformObjectAddOptionExplanation, [166](#)
  - TransformObjectCharAnnotate, [166](#)
  - TransformObjectDefineOption, [166](#)
  - TransformObjectGetByName, [166](#)
  - TransformObjectGetIntAnnotation, [167](#)
  - TransformObjectGetMark, [167](#)
  - TransformObjectGetName, [167](#)
  - TransformObjectGetSuitabilityFunction, [168](#)
  - TransformObjectGetTransformName, [168](#)
  - TransformObjectIntAnnotate, [168](#)
  - TransformObjectMark, [168](#)
  - TransformObjectSetExplanation, [169](#)
  - TransformObjectSetSuitabilityFunction, [169](#)
  - TransformObjectsGetNames, [169](#)
  - TransformObjectsMarkAll, [170](#)
  - TransformObjectsUnmarkAll, [170](#)
  - TransformObjectsUseOnly, [170](#)
  - TransformObjectUnmark, [170](#)
  - TransformReportEnabled, [171](#)
  - TransformReportTeXTable, [171](#)
  - TransformSetUserChoices, [171](#)
- TransformCharAnnotationGetIndex
  - sysprotransform.h, [140](#)
  - transform.c, [162](#)
- TransformCurrentItemDefineOption
  - sysprotransform.h, [141](#)
- TransformGetByName
  - preprocess.c, [85](#)
  - sysprotransform.h, [141](#)
- TransformGetName
  - sysprotransform.h, [141](#)
  - transform.c, [162](#)
- TransformGetNextUnmarkedItem
  - sysprotransform.h, [141](#)
  - transform.c, [163](#)
- TransformGetNItems
  - sysprotransform.h, [142](#)
- TransformGetNUnmarked
  - sysprotransform.h, [142](#)

- transform.c, 163
- TransformGetObjects
  - sysprotransform.h, 142
  - transform.c, 163
- TransformGetUserChoices
  - sysprotransform.h, 142
  - transform.c, 163
- TransformIntAnnotationGetIndex
  - sysprotransform.h, 142
  - transform.c, 164
- TransformItemCharAnnotationGetIndex
  - sysprotransform.h, 143
- TransformItemDefineOption
  - sysprotransform.h, 143
- TransformItemDescribeLong
  - sysprotransform.h, 143
  - transform.c, 164
- TransformItemDescribeShort
  - sysprotransform.h, 143
  - transform.c, 164
- TransformItemGetCharAnnotation
  - sysprotransform.h, 143
- TransformItemGetFirstOption
  - sysprotransform.h, 144
  - transform.c, 164
- TransformItemGetIntAnnotation
  - sysprotransform.h, 144
- TransformItemGetNextOption
  - sysprotransform.h, 144
  - transform.c, 165
- TransformItemGetNOptions
  - sysprotransform.h, 144
- TransformItemGetOptionI
  - sysprotransform.h, 144
- TransformItemIntAnnotate
  - sysprotransform.h, 144
- TransformItemOptionMark
  - sysprotransform.h, 145
  - transform.c, 165
- TransformItemOptionsUseOnly
  - sysprotransform.h, 145
  - transform.c, 165
- TransformObjectAddOption
  - sysprotransform.h, 145
  - transform.c, 165
- TransformObjectAddOptionExplanation
  - sysprotransform.h, 145
  - transform.c, 166
- TransformObjectCharAnnotate
  - sysprotransform.h, 146
  - transform.c, 166
- TransformObjectDefineOption
  - sysprotransform.h, 146
  - transform.c, 166
- TransformObjectGetByName
  - sysprotransform.h, 146
  - transform.c, 166
- TransformObjectGetIntAnnotation
  - sysprotransform.h, 146
  - transform.c, 167
- TransformObjectGetMark
  - sysprotransform.h, 147
  - transform.c, 167
- TransformObjectGetName
  - sysprotransform.h, 147
  - transform.c, 167
- TransformObjectGetSuitabilityFunction
  - sysprotransform.h, 147
  - transform.c, 168
- TransformObjectGetTransformName
  - sysprotransform.h, 148
  - transform.c, 168
- TransformObjectIntAnnotate
  - sysprotransform.h, 148
  - transform.c, 168
- TransformObjectMark
  - sysprotransform.h, 148
  - transform.c, 168
- transformobjects
  - SalsaTransform\_, 23
- TransformObjectSetExplanation
  - sysprotransform.h, 148
  - transform.c, 169
- TransformObjectSetSuitabilityFunction
  - sysprotransform.h, 149
  - transform.c, 169
- TransformObjectsGetNames
  - sysprotransform.h, 149
  - transform.c, 169
- TransformObjectsMarkAll
  - sysprotransform.h, 149
  - transform.c, 170

- TransformObjectsUnmarkAll
  - sysprotransform.h, 149
  - transform.c, 170
- TransformObjectsUseOnly
  - sysprotransform.h, 150
  - transform.c, 170
- TransformObjectUnmark
  - sysprotransform.h, 150
  - transform.c, 170
- TransformReportEnabled
  - sysprotransform.h, 150
  - transform.c, 171
- TransformReportTeXTable
  - sysprotransform.h, 150
  - transform.c, 171
- TransformSetUserChoices
  - sysprotransform.h, 151
  - transform.c, 171
- TRUTH
  - syspro.h, 103
- TYPELEN
  - options.c, 66
- u1.c, 172
  - copy, 172
  - main, 172
- u12.c, 172
  - main, 173
  - solvelinear, 173
- u13.c, 173
  - create\_solver, 174
  - destroy\_solver, 174
  - destroysolution, 174
  - main, 174
  - setup\_pc, 175
  - setup\_pc\_choices, 175
  - solvelinear, 175
- u14.c, 175
  - create\_solver, 176
  - destroy\_solver, 176
  - main, 176
  - setup\_pc, 177
  - setup\_pc\_choices, 177
  - solvelinear, 177
  - unset\_pc, 177
- u15.c, 178
  - create\_solver, 179
  - destroy\_solver, 179
  - main, 179
  - setup\_pc, 179
  - setup\_pc\_choices, 180
  - solvelinear, 180
  - unset\_pc, 180
- u16.c, 180
  - create\_solver, 181
  - destroy\_solver, 181
  - main, 181
  - setup\_pc, 182
  - setup\_pc\_choices, 182
  - solvelinear, 182
  - unset\_pc, 182
- u2.c, 183
  - main, 183
  - solvebycopy, 183
- u3.c, 184
  - add, 184
  - declareadders, 184
  - delprob, 185
  - delsol, 185
  - main, 185
  - makesol, 185
  - solvebycopy, 185
  - unadder, 186
- u4.c, 186
  - add, 187
  - declareadders, 187
  - delintctx, 187
  - delprob, 187
  - delsol, 187
  - main, 188
  - makeintctx, 188
  - makesol, 188
  - solvebycopy, 188
  - unadder, 189
- unadder
  - u3.c, 186
  - u4.c, 189
- unapproximate\_system
  - approximating.c, 35
- undistribute\_system
  - distribution.c, 40
- unscale\_system

---

- scaling.c, [96](#)
- unset
  - SystemPreprocessor\_, [32](#)
- unset\_ksp
  - ksp.c, [46](#)
- unset\_kspcs
  - ksp.c, [46](#)
- unset\_pc
  - pc.c, [70](#)
  - u14.c, [177](#)
  - u15.c, [180](#)
  - u16.c, [182](#)
- unsetpreprocessor
  - preprocess.c, [87](#)
- userchoices
  - SalsaTransform\_, [24](#)