1993

Rowdlas User's Guide

H. Byun

Elias N. Houstis

\textit{Purdue University}, enh@cs.purdue.edu

E. A. Vavalis

Report Number:

93-017

---

http://docs.lib.purdue.edu/cstech/1035

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.
Abstract

This report is a user's guide to ROWDLAS, a Row-oriented Distributed Linear Algebra Subroutine package. ROWDLAS can be used to develop higher level linear algebra blocks (like direct and iterative solvers) on distributed multiprocessor systems. It also provides portability across several distributed-memory multiprocessors and execution tracing for monitoring the performance and for debugging purposes.

1. Introduction. ROWDLAS is a collection of FORTRAN routines that perform basic linear algebra operations on distributed memory systems. They assume that matrices are row distributed among processors. Only information concerning the data distribution, and the interconnection topology, is required while all other architectural details are hide from the user.

For both efficiency and portability ROWDLAS is built on top of BLAS [7], [8], [2] and PICL [4] BLAS routines are used at the processor level to perform local linear algebra operations and PICL high-level communication routines are used to accumulate global information. Build-in execution tracing for monitoring the performance is available through PICL and PARAGRAPH [5]. Some preliminary data on the performance evaluation of ROWDLAS routines, together with the detailed description of the parallel algorithms used, can be found in [1].

Some of the ROWDLAS routines assume certain distribution, mapping and storage schemes for the data they involve. This information can be found in section 2. In section 3 we give detailed descriptions of the ROWDLAS routines and their arguments.

2. Assumptions and requirements. Parallel distributed memory algorithms and routines are based on certain architectural parameters like the number of processors, the interconnection topology, the numbering of the node processors and the mapping of the data onto the architecture. In ROWDLAS we have tried to make the definition of these parameters as easy as possible. In this section we list the available choices for these parameters and describe the mechanisms to assign values to them.

Before the call of any of the ROWDLAS routines the users should:

- Call the PICL routine open() to specify the number of processors nprocs, the host and node ids, own and host respectively and enable communication. These parameters should be stored in the common block
  
  common /open/ nprocs, own, host

- Call the PICL routine setare0() to specify the interconnection topology and the numbering of the nodes. setare0 can be called more than once in case one needs to modify the topology or the node numbering according to the needs of the ROWDLAS routines.

- Some routines require information on the distribution of the data. In this case the user should provide the array idst. idst(i) \(i = 1, \ldots, nprocs + 1\) hold the global index of the first element of the data that belongs to processor \(i\).
In the following table we list the possible choices for the above above mentioned parameters for each ROWDLAS routine.

The user simply links the ROWDLAS, and PCL libraries together with his favored uni-processor BLAS routines, like the highly optimized ones in [6].

3. Routine Descriptions. In this section we describe the Linear Algebra Subprograms currently available in ROWDLAS. We use the BLAS naming convention and the argument lists are essential the BLAS ones with the only exception of the idst array that hold the data distribution information.

The routines are arranged in alphabetical order and all of them come in single and double precision real.
dsamin/ddamin

Syntax: \( w = \text{dsamin/ddamin}(n,x,\text{idst}) \)

Purpose: Finds the smallest (in absolute value) element of the vector \( x \).

\[
    w = \min_{1 \leq i \leq n} (|x_i|)
\]

On Entry

\( n \) integer

the order of the vector \( x \).

\( x \) Single precision real for dsamin
Double precision real for ddamin

array of dimension \( n \)

\( \text{idst} \) integer array of dimension \( nprocs \)

array holding the distribution information of \( x \).

On Return

\( w \) single precision real for dsamin
single precision real for ddamin

the absolute value of the smallest (in absolute value) element of \( x \).
dsamax/ddamax

Syntax: \( w = \text{dsamax/ddamax}(n,x,\text{idst}) \)

Purpose: Finds the largest (in absolute value) element of the vector \( x \).
\[
 w = \max (|x_i|) 
\]

On Entry
\begin{itemize}
  \item \( n \) integer
    \begin{itemize}
      \item the order of the vector \( x \).
    \end{itemize}
  \item \( x \) Single precision real for dsamax
    \begin{itemize}
      \item Double precision real for ddamax
    \end{itemize}
    \begin{itemize}
      \item array of dimension \( n \)
    \end{itemize}
  \item \( \text{idst} \) integer array of dimension \( nprocs \)
    \begin{itemize}
      \item array holding the distribution information of \( x \).
    \end{itemize}
\end{itemize}

On Return
\begin{itemize}
  \item \( w \) single precision real for dsamax
    \begin{itemize}
      \item double precision real for ddamax
    \end{itemize}
    \begin{itemize}
      \item the absolute value of the largest (in absolute value) element of \( x \).
    \end{itemize}
\end{itemize}
idsamin/iddamin

Syntax: \( iw = \text{idsamin/iddamin}(n,x,\text{idst}) \)

Purpose: Finds the smallest index \( i \) such that:
\[
|x_i| = \min_{1 \leq j \leq n} |x_j|, j = 1 \ldots n
\]

On Entry

\( n \) integer
the order of the vector \( x \).

\( x \) Single precision real for idsamin
Double precision real for iddamin
array of dimension \( n \)

\( \text{idst} \) integer array of dimension \( nproes \)
array holding the distribution information of \( x \).

On Return

\( iw \) integer
index of smallest (in absolute value) element of \( x \).
idsamax/iddamax

Syntax: \( \text{iw} = \text{idsamax/iddamax}(\text{n}, \text{x}, \text{idst}) \)

Purpose: Finds the largest index \( i \) such that:
\[
|x_i| = \max_{1 \leq j \leq n} |x_j|, j = 1, \ldots, n
\]

On Entry
\( n \) integer
the order of the vector \( x \).
\( x \) Single precision real for idsamax
Double precision real for iddamax
array of dimension \( n \)
\( \text{idst} \) integer array of dimension \( nprocs \)
array holding the distribution information of \( x \).

On Return
\( \text{iw} \) integer
index of largest (in absolute value) element of \( x \).
**dsasum/ddasum**

**Syntax:** \( w = \text{dsasum/ddasum}(n,x,\text{idst}) \)

**Purpose:** Computes the sum of magnitudes of the elements of the vector \( x \):

\[
    w = \sum_{i=1}^{n} |x_i|
\]

**On Entry**

\( n \)  
integer

the order of the vector \( x \).

\( x \)  
Single precision real for dsasum  
Double precision real for ddasum

array of dimension \( n \).

\( \text{idst} \)  
integer array of dimension \( nprocs \)

array holding the distribution information of \( x \).

**On Return**

\( w \)  
Single precision real for dsasum  
Double precision real for ddasum

sum of magnitudes of the elements of \( x \).
dsnrm2/ddnrm2

Syntax:  \( w = \text{dsnrm2/ddnrm2}(n,x,\text{idst}) \)

Purpose: Computes the Euclidean norm of the vector \( x \):

\[
w = \left( \sum_{i=1}^{n} x_i^2 \right)^{1/2}
\]

On Entry

\( n \)  
integer
the order of the vector \( x \).

\( x \)  
Single precision real for dsnrm2
Double precision real for ddnrm2
array of dimension \( n \)

\( \text{idst} \)  
integer array of dimension \( nprocs \)
array holding the distribution information of \( x \).

On Return

\( w \)  
single precision real for dsnrm2
double precision real for ddnrm2
Euclidean norm of \( x \).
**dsdot/dddot/dddsdot**

Syntax: \( w = \text{dsdot/dddot/dddsdot}(n,x,y,idst) \)

**Purpose:** Computes the dot product of two vectors:
\[ w = x^T y \]

ddsdot takes single precision arguments, but performs the summation in double precision and returns a double precision result.

**On Entry**

\( n \) integer

the order of the vector \( x \).

\( x \) LOCAL single precision real for dsdot and ddsdot
LOCAL double precision real for dddot

array of dimension \( n \)

\( y \) Single precision real for dsdot and ddsdot
Double precision real for dddot

array of dimension \( n \)

\( idst \) integer array of dimension \( nprocs \)

array holding the distribution information of \( y \).

**On Return**

\( w \) single precision real for dsdot
double precision real for dddot and ddsdot

the dot product of \( x \) and \( y \).
dsgemv/ddgemv

Syntax: \( w = \text{dsgemv/ddgemv}(\text{trans}, \alpha, \beta, a, \text{lda}, x, \text{beta}, \text{idst}) \)

Purpose: Performs one of the matrix-vector operations

\[
y = \alpha A x + \beta y
\]

\[
y = \alpha A^T x + \beta y
\]

where \( \alpha \) and \( \beta \) are scalars, \( x \) and \( y \) are vectors and \( A \) is an \( m \) by \( n \) matrix.

On Entry

- \text{trans} character"1"
- \text{alpha} Single precision for dsgemv
- \text{beta} Single precision for dsgemv
- \text{lda} integer
- \text{n} integer
- \text{m} integer

the number of rows of the matrix \( a \).

the number of columns of the matrix \( a \).

specifies the scalar \( \alpha \)

array of dimension \( (\text{lda}, n) \). Contains the matrix on its leading \( m \) by \( n \) part

the first dimension of \( a \) as declared in the calling (sub)program. must be at least \( \max(1, m) \)

array of dimension \( n \) if \text{trans} = \char92 n \) or \( m \) if \text{trans} = \char92 t \).

10
beta 
Single precision for dsgemv
Double precision for ddgemv

specifies the scalar $\beta$

$y$
Single precision for dsgemv
Double precision for ddgemv

array of dimension $n$ if $\text{trans} = 'n'$ or $m$ if $\text{trans} = 't'$.

idst
integer array of dimension $nprocs$

array holding the distribution information of $A, x$ and $y$.

On Return
$y$
overwritten by the updated vector $y$. 
dsbemv/ddbemv

Syntax:  \( w = \text{dsbemv/ddbemv}(\text{trans}, m, n, kl, ku, \alpha, a, \text{lda}, x, \beta, y, \text{idst}) \)

Purpose: Performs one of the matrix-vector operations

\[
y = \alpha A x + \beta y
\]

\[
y = \alpha A^T x + \beta y
\]

where \( \alpha \) and \( \beta \) are scalars, \( x \) and \( y \) are vectors and \( A \) is an \( m \) by \( n \) banded matrix with lower bandwidth \( kl \) and upper bandwidth \( ku \).

On Entry

- \( \text{trans} \) character*1
  - specifies the operation to be performed as follows
    - \( \text{trans} = \text{'N'} \) or \( \text{'n'} \) \( y = \alpha A x + \beta y \)
    - \( \text{trans} = \text{'T'} \) or \( \text{'t'} \) \( y = \alpha A^T x + \beta y \)

- \( m \) integer
  - the number of rows of the matrix \( a \).

- \( n \) integer
  - the number of columns of the matrix \( a \).

- \( kl \) integer
  - the number of sub-diagonals of the matrix \( a \).

- \( ku \) integer
  - the number of super-diagonals of the matrix \( a \).

- \( \alpha \) Single precision for dsbemv
  - Double precision for ddbemv
  - specifies the scalar \( \alpha \)

- \( a \) Single precision for dsbemv
  - Double precision for ddbemv
  - array of dimension (\( \text{lda}, n \)). Contains the matrix on its leading \( m \) by \( n \) part
  - The matrix is locally stored in standard (not banded) form
lda

integer

the first dimension of a as declared in the calling (sub)program.
must be at least \( \max(1, m) \)

x

Single precision for dsbenv
Double precision for ddbenv

array of dimension \( n \) if \( \text{trans} = 'u' \) or \( m \) if \( \text{trans} = 'l' \).

beta

Single precision for dsbenv
Double precision for ddbenv

specifies the scalar \( \beta \)

y

Single precision for dsbenv
Double precision for ddbenv

array of dimension \( n \) if \( \text{trans} = 'u' \) or \( m \) if \( \text{trans} = 'l' \).

idst

integer array of dimension \( nprocs \)

array holding the distribution information of \( A, x \) and \( y \).

On Return

\( y \)

overwritten by the updated vector \( y \).

NOTES

- On each processor, only the non-zero \( \text{BLOCKS} \) of the matrix \( A \) are stored.
- The distributed data should be mapped assuming ring or full connectivity and the \text{setarcc0} should be called accordingly.
dssemv/ddsemv

Syntax: w = dssemv/ddsemv(trans.m.n.alpha.coef.idcoef.lda.x.beta.y.idst)

Purpose: Performs one of the matrix-vector operations

\[ y = \alpha Ax + \beta y \]

\[ y = \alpha A^T x + \beta y \]

where \( \alpha \) and \( \beta \) are scalars, \( x \) and \( y \) are vectors and \( A \) is an \( m \) by \( n \) sparse matrix stored in a compressed form using the arrays coef and idcoef.

On Entry

trans character*1

specifies the operation to be performed as follows

trans = 'N' or 'n' \( y = \alpha Ax + \beta y \)

trans = 'T' or 't' \( y = \alpha A^T x + \beta y \)

\( m \) integer

the number of rows of the matrix \( a \).

\( n \) integer

the number of columns of the matrix \( a \).

alpha Single precision for dssemv

Double precision for ddsemv

specifies the scalar \( \alpha \)

coef Single precision for dssemv

Double precision for ddsemv

array of dimension \((lda,n)\).

Contains the coefficients of the matrix \( A \) on its leading \( m \) by \( n \) part.

idcoef Single precision for dssemv

Double precision for ddsemv

array of dimension \((lda,n)\).

Contains the indices of the coefficients of the matrix \( A \) on its leading \( m \) by \( n \) part.
lda  
integer  
the first dimension of coef and ideof as declared in the calling (sub)program,  
must be at least max(1, m)  

x  
Single precision for dssemv  
Double precision for ddsemv  
array of dimension n if trans = 'n' or m if trans = 't'.  

beta  
Single precision for dssemv  
Double precision for ddsemv  
specifies the scalar $\beta$  

y  
Single precision for dssemv  
Double precision for ddsemv  
array of dimension n if trans = 'n' or m if trans = 't'.  

idst  
integer array of dimension nprows  
array holding the distribution information of A, x and y.  

On Return  
y overwritten by the updated vector y.  

NOTES  
- The distributed data should be mapped assuming ring or full connectivity and the setarco should be called accordingly.
REFERENCES


