?gesv

Computes the solution to the system of linear equations with a square coefficient matrix A and multiple right-hand sides.

Syntax

```
lapack_int LAPACKE_sgesv (int matrix_layout , lapack_int n ,
lapack_int nrhs , float * a , lapack_int lda , lapack_int * ipiv ,
float * b , lapack_int ldb );
```

```
lapack_int LAPACKE_dgesv (int matrix_layout , lapack_int n ,
lapack_int nrhs , double * a , lapack_int lda , lapack_int * ipiv
, double * b , lapack_int ldb );
```

```
lapack_int LAPACKE_cgesv (int matrix_layout , lapack_int n ,
lapack_int nrhs , lapack_complex_float * a , lapack_int lda ,
lapack_int * ipiv , lapack_complex_float * b , lapack_int ldb );
```

```
lapack_int LAPACKE_zgesv (int matrix_layout , lapack_int n ,
lapack_int nrhs , lapack_complex_double * a , lapack_int lda ,
lapack_int * ipiv , lapack_complex_double * b , lapack_int ldb
);
```

```
lapack_int LAPACKE_dsgesv (int matrix_layout, lapack_int n,
lapack_int nrhs, double * a, lapack_int lda, lapack_int * ipiv,
double * b, lapack_int ldb, double * x, lapack_int ldx,
lapack_int * iter);
```

```
lapack_int LAPACKE_zcgesv (int matrix_layout, lapack_int n,
lapack_int nrhs, lapack_complex_double * a, lapack_int lda,
lapack_int * ipiv, lapack_complex_double * b, lapack_int ldb,
lapack_complex_double * x, lapack_int ldx, lapack_int * iter);
```

Include Files

- `mkl.h`

Description

The routine solves for *X* the system of linear equations `A*X = B`, where *A* is an *n*-by-*n* matrix, the columns of matrix *B* are individual right-hand sides, and the columns of *X* are the corresponding solutions.

The *LU* decomposition with partial pivoting and row interchanges is used to factor *A* as `A = P*L*U`, where *P* is a permutation matrix, *L* is unit lower triangular, and *U* is upper triangular. The factored form of *A* is then used to solve the system of equations `A*X = B`.

The `dsgesv` and `zcgesv` are mixed precision iterative refinement subroutines for exploiting fast single precision hardware. They first attempt to factorize the matrix in single precision (`dsgesv`) or single complex precision (`zcgesv`) and use this factorization within an iterative refinement procedure to produce a solution with double precision (`dsgesv`) / double complex precision (`zcgesv`) normwise backward error quality (see below). If the approach fails, the method switches to a double precision or double complex precision factorization respectively and computes the solution.

The iterative refinement is not going to be a winning strategy if the ratio single precision performance over double precision performance is too small. A reasonable strategy should take the number of right-hand sides and the size of the matrix into account. This might be done with a call to `ilaenv` in the future. At present, iterative refinement is implemented.

The iterative refinement process is stopped if

```
iter > itermax
```

or for all the right-hand sides:

```
rnmr < sqrt(n)*xnrm*anrm*eps*bwdmax
```

where

- `iter` is the number of the current iteration in the iterativerefinement process
- `rnmr` is the infinity-norm of the residual

For more complete information about compiler optimizations, see

- `xnrm` is the infinity-norm of the solution
- `anrm` is the infinity-operator-norm of the matrix *A*
- `eps` is the machine epsilon returned by `dlamch` (‘Epsilon’).

The values `itermax` and `bwdmax` are fixed to 30 and 1.0d+00 respectively.

Input Parameters

matrix_layout	Specifies whether matrix storage layout is row major (`LAPACK_ROW_MAJOR`) or column major (`LAPACK_COL_MAJOR`).
n	The number of linear equations, that is, the order of the matrix *A*; *n*≥ 0.
nrhs	The number of right-hand sides, that is, the number of columns of the matrix *B*; *nrhs*≥ 0.
a	The array *a*(size max(1, *lda***n*)) contains the *n*-by-*n* coefficient matrix *A*.
b	The array *b*of size max(1, *ldb***nrhs*) for column major layout and max(1, *ldb***n*) for row major layout contains the *n*-by-*nrhs* matrix of right hand side matrix *B*.
lda	The leading dimension of the array *a*; *lda*≥ max(1, *n*).
ldb	The leading dimension of the array *b*; *ldb*≥ max(1, *n*) for column major layout and *ldb*≥*nrhs* for row major layout.

ldx	The leading dimension of the array *x*; *ldx*≥ max(1, *n*) for column major layout and *ldx*≥*nrhs* for row major layout.

Output Parameters

a	Overwritten by the factors *L* and *U* from the factorization of *A* = *P**L*U*; the unit diagonal elements of *L* are not stored. If iterative refinement has been successfully used (*info*= 0 and *iter*≥ 0), then *A* is unchanged. If double precision factorization has been used (*info*= 0 and *iter* < 0), then the array *A* contains the factors *L* and *U* from the factorization *A* = *P**L*U*; the unit diagonal elements of *L* are not stored.
b	Overwritten by the solution matrix *X* for `dgesv`, `sgesv`,`zgesv`,`zgesv`. Unchanged for `dsgesv` and `zcgesv`.
ipiv	Array, size at least `max(1, n)`. The pivot indices that define the permutation matrix *P*; row *i* of the matrix was interchanged with row *ipiv*[i-1]. Corresponds to the single precision factorization (if *info*= 0 and *iter*≥ 0) or the double precision factorization (if *info*= 0 and *iter* < 0).
x	Array, size max(1, *ldx**nrhs*) for column major layout and max(1, *ldx**n*) for row major layout. If *info* = 0, contains the *n*-by-*nrhs* solution matrix *X*.

iter

If *iter* < 0: iterative refinement has failed, double precision factorization has been performed

- If *iter* = -1: the routine fell back to full precision for implementation- or machine-specific reason
- If *iter* = -2: narrowing the precision induced an overflow, the routine fell back to full precision
- If *iter* = -3: failure of `sgetrf` for `dsgesv`, or `cgetrf` for `zcgesv`
- If *iter* = -31: stop the iterative refinement after the 30th iteration.

If *iter* > 0: iterative refinement has been successfully used. Returns the number of iterations.

Return Values

This function returns a value *info*.

If *info*=0, the execution is successful.

If *info* = -*i*, parameter *i* had an illegal value.

If *info* = *i*, $U_{i,\ i}$ (computed in double precision for mixed precision subroutines) is exactly zero. The factorization has been completed, but the factor *U* is exactly singular, so the solution could not be computed.

Parent topic: LAPACK Linear Equation Driver Routines (/node/eebab4d8-106f-4afa-9a0c-744bbecd5631)

See Also

`dlamch` (/node/70b6c0a0-2e0b-4c0f-9413-2afcfc8e60d8#70B6C0A0-2E0B-4C0F-9413-2AFCFC8E60D8)
`sgetrf` (/node/e4779e02-346c-4670-92ab-c67bd8559051#E4779E02-346C-4670-92AB-C67BD8559051)

For more complete information about compiler optimizations, see

Matrix Storage Schemes (/node/dc524afc-82dd-421e-868a-f40388eb7826#DC524AFC-82DD-421E-868A-F40388EB7826)

For more complete information about compiler optimizations, see our Optimization Notice (/en-us/articles/optimization-notice#opt-en).