LAPACK quick reference guide

Prefixes

Each routine has a prefix, denoted by a hypen - in this guide, made of 3 letters *xyy*, where *x* is the data type, and *yy* is the matrix type.

Data type			
s	single	d	double
c	complex single	z	complex double

Matrix type	full	banded	packed	tridiag	generalized problem
general	ge	gb		gt	gg
symmetric	sy	sb	sp	st	
Hermitian	he	hb	hp		
SPD / HPD	po	pb	pp	pt	
triangular	tr	tb	tp		tg
upper Hessenberg	hs				hg
trapezoidal	tz				
orthogonal	or		op		
unitary	un		up		
diagonal	di				
bidiagonal	bd				

For symmetric, Hermition, and triangular matrices, elements below/above the diagonal (for upper/lower respectively) are not accessed. Similarly for upper Hessenberg, elements below the subdiagonal are not accessed.

Packed storage is by columns. For example, a 3×3 upper triangular is stored as

$$\left[\underbrace{a_{11}} \; \underbrace{a_{12} \;\; a_{22}} \; \underbrace{a_{13} \;\; a_{23} \;\; a_{33}} \right]$$

Banded storage puts columns of the matrix in corresponding columns of the array, and diagonals in rows of the array, for example:

$$\begin{array}{l} \left[\begin{array}{ccccc} * & a_{12} & a_{23} & a_{34} & a_{45} \\ a_{11} & a_{22} & a_{33} & a_{44} & a_{55} \\ a_{21} & a_{32} & a_{43} & a_{54} & * \\ a_{31} & a_{42} & a_{53} & * & * \end{array}\right] \end{array} \begin{array}{l} \text{1st diagonal} \\ \text{2nd (main) diagonal} \\ \text{3rd diagonal} \\ \text{4th diagonal} \end{array}$$

Bi- and tridiagonal matrices are stored as 2 or 3 vectors of length n and $n-1$.

Updated March 19, 2008.
BLAS and LAPACK guides available from http://www.ews.uiuc.edu/˜mrgates2/.
Reference: *LAPACK Users Guide* from http://www.netlib.org/lapack/faq.html
Copyright ©2007–2008 by Mark Gates. This document is free; you can redistribute it under terms of the GNU General Public License, version 2 or later.

Drivers

Drivers are higher level routines that solve an entire problem.

Linear system, solve $Ax = b$.
`-sv` — solve
`-svx` — expert; also $A^T x = b$ or $A^H x = b$, condition number, error bounds, scaling
Matrix types [General ge, gb, gt; SPD po, pp, pb, pt; Symmetric sy, sp, he, hp]

Linear least squares, minimize $\|b - Ax\|_2$.
`-ls` — full rank, rank$(A) = \min(m,n)$, uses QR.
`-lsy` — rank deficient, uses complete orthogonal factorization.
`-lsd` — rank deficient, uses SVD.
Matrix types [General ge]

Generalized linear least squares.
Minimize $\|c - Ax\|_2$ subject to $Bx = d$.
`-lse` — B full row rank, matrix $\begin{bmatrix} A \\ B \end{bmatrix}$ full col rank.

Minimize $\|y\|_2$ subject to $d = Ax + By$.
`-glm` — A full col rank, matrix $\begin{bmatrix} A & B \end{bmatrix}$ full row rank.
Matrix types [General gg]

Eigenvalues, solve $Ax = \lambda x$.
Symmetric
`-ev` — all eigenvalues, [eigenvectors]
`-evx` — expert; also subset
`-evd` — divide-and-conquer; faster but more memory
`-evr` — relative robust; fastest and least memory
Matrix types [Symmetric sy, sp, sb, st, he, hp, hb]

Nonsymmetric
`-ev` — eigenvalues, [left, right eigenvectors]
`-evx` — expert; also balance matrix, condition numbers
`-es` — Schur factorization
`-esx` — expert; also condition numbers
Matrix types [General ge]

Generalized eigenvalue, solve $Ax = \lambda Bx$
Symmetric, B SPD
`-gv` — all eigenvalues, [eigenvectors]
`-gvx` — expert; also subset
`-gvd` — divide-and-conquer, faster but more memory
Matrix types [Symmetric sy, sp, sb, he, hp, hb]

Nonsymmetric
`-ev` — eigenvalues, [left, right eigenvectors]
`-evx` — expert; also balance matrix, condition numbers
`-es` — Schur factorization
`-esx` — expert; also condition numbers
Matrix types [General gg]

SVD singular value decomposition, $A = U\Sigma V^H$
`-svd` — singular values, [left, right vectors]
`-sdd` — divide-and-conquer; faster but more memory
Matrix types [General ge]

Generalized SVD, $A = U\Sigma_1 Q^T$ **and** $B = V\Sigma_2 Q^T$
`-svd` — singular values, [left, right vectors]
Matrix types [General gg]

Computational routines

Computational routines perform one step of solving the problem. Drivers call a sequence of computational routines.

Triangular factorization

`-trf` — factorize: General LU, Cholesky LL^T, tridiag LDL^T, sym. indefinite LDL^T
`-trs` — solve using factorization
`-con` — condition number estimate
`-rfs` — error bounds, iterative refinement
`-tri` — inverse (not for band)
`-equ` — equilibrate A (not for tridiag, symmetric indefinite, triangular)
Matrix types [General ge, gb, gt; SPD po, pp, pb, pt; Symmetric sy, sp, he, hp, Triangular tr, tb]

Orthogonal factorization

`-qp3` — QR factorization, with pivoting
`-qrf` — QR factorization
`-rqf` — RQ factorization
`-qlf` — QL factorization
`-lqf` — LQ factorization
`-tzrqf` — RQ factorization
`-tzrzf` — RZ trapezoidal factorization
Matrix types [General ge, some Trapezoidal tz]

`-gqr` — generate Q after `-qrf`
`-grq` — generate Q after `-rqf`
`-gql` — generate Q after `-qlf`
`-glq` — generate Q after `-lqf`
`-mqr` — multiply by Q after `-qrf`
`-mrq` — multiply by Q after `-rqf`
`-mql` — multiply by Q after `-qlf`
`-mlq` — multiply by Q after `-lqf`
`-mrz` — multiply by Q after `-tzrzf`
Matrix types [Orthogonal or, un]

Generalized orthogonal factorization

`-qrf` — QR of A, then RQ of $Q^T B$
`-rqf` — RQ of A, then QR of BQ^T
Matrix types [General gg]

Eigenvalue

`-trd` — tridiagonal reduction
Matrix types [Symmetric sy, he, sp, hp]

`-gtr` — generate matrix after `-trd`
`-mtr` — multiply matrix after `-trd`
Matrix types [Orthogonal or, op, un, up]

Symmetric tridiagonal eigensolvers
`-eqr` — using implicitly shifted QR
`-pteqr` — using Cholesky and bidiagonal QR
`-erf` — using square-root free QR
`-edc` — using divide-and-conquer
`-egr` — using relatively robust representation
`-ebz` — eigenvalues using bisection
`-ein` — eigenvectors using inverse iteration
Matrix types [Symmetric tridiag st, one SPD pt]

Nonsymmetric
`-hrd` — Hessenberg reduction
`-bal` — balance
`-bak` — back transforming
Matrix types [General ge]

`-ghr` — generate matrix after `-hrd`
`-mhr` — multiply matrix after `-hrd`
Matrix types [Orthogonal or, un]

`-eqr` — Schur factorization
`-ein` — eigenvectors using inverse iteration
Matrix types [upper Hessenberg hs]

`-evc` — eigenvectors
`-exc` — reorder Schur factorization
`-syl` — Sylvester equation
`-sna` — condition numbers
`-sen` — condition numbers of eigenvalue cluster/subspace
Matrix types [Triangular tr]