IMPROVED DORGQR AND RECURSIVE DORG2R AGAINST OPTIMIZED BLAS ROUTINES *

JOHNATHAN RHYNE †

In collaboration with: Julien Langou Advisor

Abstract. The LAPACK routine dorg2r is used to attain the full Q matrix in the QR decomposition, and is needed even when the blocked version is used. Using a recursive scheme, we improve the performance of the dorg2r in the case of having number of reflectors equal to the number of columns of Q. In addition, our scheme produces a way to compute The first k columns of Q only, which can be useful when coupled with a more efficient scheme to produce the remaining columns needed. We also propose a new version of dorgqr where we also see performance increases for modestly sized inputs against both reference LAPACK and tested optimized BLAS routines.

Key words. lapack blas recursive qr nullspace

1. Introduction. The lapack routines `dorgqr` and `dorg2r` are two methods of computing the Q matrix associated with the QR decomposition of a matrix, denoted A, from the householder reflectors. The existing functionality in LAPACK does not properly take advantage of the fact that our first step in computing the columns between the desired number of columns of Q, denoted n, and the number of reflectors used in the decomposition of A, denoted k, involves a matrix multiplication where one of the elements is the identity matrix. We exploit this fact.

2. Main Contribution. Our main contribution is an implementation of an improved `dorgqr` . We do this by taking advantage of the first step in the `dorgqr` algorithm being a matrix multiplication by an identity matrix and blocking for the entire matrix instead of calling `dorg2r` to do these first columns.

3. Numerical Results. Our numerical experiments test on the following hardware:
`Insert processor information for laptop and the HPC.`
We test 3 different versions of `dorgqr` .

Version 1: Updated `dorgqr`
Version 2: Adding our new recursive `dorgkr` as a helper routine
Version 3: Adding our new recusrive `dlarft` as a helper routine

For Version 2, we also test `dorgkr` against `dorg2r` for performance that mimic the instances it is called from within dorgqr, so that we can demonstrate the gain directly. This means that we do not time the call to `dlarft` as this call is done regardless inside `dorgqr` .

For Version 3, we perform the same timing as for Version 1, but also include timing just the calls to dlarft.

Our testing paradigm is as follows:

We test only accuracy with the fortran tester routines. We have these Fortran testers to be able to see how accurate we are under these conditions. However, we have a tester that lives only in `C` that calls the fortran subroutines to measure both accuracy and timing. We treat these as the main source for this write-up as it is the

*This work was funded by IJK.
†University of Colorado Denver, (johnathan.rhyne@ucdenver.edu).

closest related to how most users will interact with the LAPACK library.

4. Conclusions.