Tuning NEC with a faster LU

Jürgen v. Hagen
Institut für Höchstfrequenztechnik und Elektronik
Universität Karlsruhe, Kaiserstr. 12, D - 76128 Karlsruhe, Germany,
Phone: +49 721 608 76 76, Fax: +49 721 69 18 65
email: vonhagen@ihe.etec.uni-karlsruhe.de

Abstract

The well-known and widely used NEC code in its forms NEC 2 and NEC 4 is affected by a rather slow *LU* factorization routine. It is shown how few small changes to the code speed up the solution process considerably, almost by two orders of magnitude.

I. Introduction

NEC 2 and NEC 4 are widely used codes for computing radiation pattern, scattering problems etc. Whereas NEC 4 is still affected by a serious drawback (the code cannot be used outside the US), NEC 2 has a wide user community. However, in the time NEC was developed, computers have not had the capabilities of today's processors. Still, small changes (166 lines) can be made to NEC to improve the performance.

The modifications use the mathematical routines in LAPACK [1], [2] and the standardized subroutines in [3], [4], [5], [6]. Further optimization was possible by efficient cache re-use and has been reported in [7], [8], [9], [10], The above numerical subroutines and the optimization have found their way into the mathematical subroutines and libraries from almost all vendors of computer platforms and are available either commercially or at no cost.

In the following, the modifications to the `nec2d` sources are described to obtain the new `nec2j`, where optimized libraries can be found, and finally what performance improvements you can expect.

II. Modifications to the NEC sources

Several changes are necessary to modify the `nec2d` sources. Beside the modifications to the subroutines that effectively call the appropriate algorithms, modifications to the parser are necessary to switch between the original and the new *LU* factorization routines. Therefore, some minor changes are necessary to the input parser that reads the NEC-input files. Finally, some modifications were necessary to compile on certain platforms (g77 on Linux) that are of minor importance.

First, modifications to the parser include the creation of a new `MS`-card that switches between the original and the new *LU* factorization routines. Fig. 2(a) show the changes (lines that begin by `<` are the old lines in `nec2d`, lines with `>` are the new lines for `nec2j`). A new common block includes the new card and a definition for two variables that are switches for the algorithm `msolver` (0 for the original algorithm) and an optional computation of an estimate for the condition number `compcn` (0 for no computation). Default is solution by the optimized algorithm and no condition estimate. Changes in the subroutines `FACTRS` and `SOLVES` are shown in Figs. 1(a) and 1(b), respectively.

Finally, some minor modifications that were necessary to compile on certain platforms are shown in Fig. 2(b).

III. Where to find optimized BLAS and LAPACK routines

To take advantage of the optimized subroutines, libraries are needed. They can be obtained (in the easiest case) from the vendor of the platform. Vendors, the name of the library and a first Internet-site are shown in table I. For almost any platform, auto-optimizing subroutines are available [11], [12]. For using the Atlas library you need a C compiler and a Fortran compiler. Some precompiled libraries are available.

In any case, you need to modify the sources, compile the sources and link together with the libraries above in order to obtain the high performance NEC.

IV. Example Performances

A sample problem with 2096 unknowns is reported in table II. The two platforms are Hewlett-Packard PA Risc processors with the HP UX 10.20 and 11.00. The vendor's `mlib` was used together with the HP compiler f90. The second platform is an Athlon running Linux and the Atlas library. The Gnu compiler gcc 2.95 together with g77 was used.

V. Final Comments

Due to the unclear copyright situation, I will not distribute the modified sources. I have not tried any platform other than the above two cited platforms. Performance improvements may be considerably different on your platform.

References

[1] E. Anderson, Z. Bai, J. Dongarra, A. Greenbaum, A. McKenney, J. D. Croz, S. Hammarling, J. Demmel, C. Bischof, and S. Sorenson, "LAPACK: A portable linear algebra library for high-performance computers," *Proceedings of Supercomputing '90*, pp. 2–11, Nov. 1990.

```
28a29,31
>       INTEGER msolver, compcn
>       COMMON /MSOLVERPAR/ msolver, compcn
>       SAVE /MSOLVERPAR/
67c70
<       DIMENSION ATST(22),PNET(6),HPOL(3),IX(2*MAXSEG)
---
>       DIMENSION ATST(23),PNET(6),HPOL(3),IX(2*MAXSEG)
77c80
<      1 'NX','EN','TL','PT','KH','NH','PQ','EK','WG','CP','PL'/
---
>      1 'NX','EN','TL','PT','KH','NH','PQ','EK','WG','CP','PL','MS'/
85a89,91
>       write(*,*) ' NEC 2 - LAPACK version '
>       write(*,"('compiled for ',I5,' segments')") MAXSEG
>       write(*,"('maximum matrix size ',I5)") MAXMAT
177a184,186
>       msolver = 1
>       compcn = 0
> C***
230a240
>       IF (AIN.EQ.ATST(23)) GO TO 331
423a434,442
> C***
> C
> C     SOLVER FLAGS
> C
> 331   continue
>       if (ITMP1 .EQ. 0) msolver = 0
>       if ((ITMP1 .NE. 0) .AND. (ITMP2 .NE. 0)) compcn = 1
> C***
>       GO TO 14
3658c3677,3690
<       DIMENSION A(1), IP(NROW), IX(NROW)
---
>       DIMENSION A(*), IP(NROW), IX(NROW)
> C     additional variables for NETLIB LU and condition number
> C     juergen v.Hagen 1999
>       integer info
>       real*8 anorm
>       include "NEC2DPAR.INC"
>       complex*16 work(2*MAXSEG)
>       real*8 rwork(2*MAXSEG)
>
>       integer msolver, compcn
>       common /msolverpar/ msolver, compcn
>       save /msolverpar/
> C***
>
3663c3695,3712
< 1     CALL FACTR (NP,A(KA),IP(KA),NROW)
---
>       if (msolver .eq. 0) then
>       CALL FACTR (NP,A(KA),IP(KA),NROW)
> C     netlib LU
>       else if (msolver .eq. 1) then
> C     compute condition number
>         if (compcn .eq. 1) then
>            anorm = zlange("1", NP, NP, A(KA), nrow, work)
>         endif
>         call zgetrf (np, np, A(KA), nrow, IP(KA), info)
>         if (info .ne. 0) print *,'ZGETRF info=',info
> C     condition number second part
>         if (compcn .eq. 1) then
>            call zgecon("1", NP, A(KA), NROW, anorm, condnum,
>      &                 work, rwork, info)
>            WRITE(3,'(20X,"CONDITION NUMBER ",G)') 1.0d0/condnum
>         endif
>       endif
> 1     continue
8336c8385
<       DIMENSION A(1), B(N1C,1), C(N1C,1), D(N2CZ,1), IP(1), XY(1)
---
>       DIMENSION A(*), B(N1C,*), C(N1C,*), D(N2CZ,*), IP(*), XY(*)
8507c8556,8561
<       DIMENSION A(1), IP(1), B(NEQ,NRH)
---
>       DIMENSION A(*), IP(*), B(NEQ,NRH)
>       integer info
>       integer msolver, compcn
>       common /msolverpar/ msolver, compcn
>       save /msolverpar/
>
8567c8621,8626
< 14    CALL SOLVE (NPEQ,A(IB),IP(IA),B(IA,IC),NROW)
---
>         if (msolver .eq. 0) then
>           CALL SOLVE (NPEQ,A(IB),IP(IA),B(IA,IC),NROW)
>         else if (msolver .eq. 1) then
>           call zgetrs("T",NPEQ,NRH,A(IB),nrow,IP(IA),B(IA,IC),NROW,info)
>         end if
> 14    continue
```

(a) Changes to the nec2d code for using the optimized routines - changes to subroutine `FACTRS`.


```
28a29,31
>       INTEGER msolver, compcn
>       COMMON /MSOLVERPAR/ msolver, compcn
>       SAVE /MSOLVERPAR/
67c70
<       DIMENSION ATST(22),PNET(6),HPOL(3),IX(2*MAXSEG)
---
>       DIMENSION ATST(23),PNET(6),HPOL(3),IX(2*MAXSEG)
77c80
<      1 'NX','EN','TL','PT','KH','NH','PQ','EK','WG','CP','PL'/
---
>      1 'NX','EN','TL','PT','KH','NH','PQ','EK','WG','CP','PL','MS'/
85a89,91
>       write(*,*) ' NEC 2 - LAPACK version '
>       write(*,"('compiled for ',I5,' segments')") MAXSEG
>       write(*,"('maximum matrix size ',I5)") MAXMAT
177a184,186
>       msolver = 1
>       compcn = 0
> C***
230a240
>       IF (AIN.EQ.ATST(23)) GO TO 331
423a434,442
> C***
> C
> C     SOLVER FLAGS
> C
> 331   continue
>       if (ITMP1 .EQ. 0) msolver = 0
>       if ((ITMP1 .NE. 0) .AND. (ITMP2 .NE. 0)) compcn = 1
> C***
>       GO TO 14
3658c3677,3690
<       DIMENSION A(1), IP(NROW), IX(NROW)
---
>       DIMENSION A(*), IP(NROW), IX(NROW)
> C     additional variables for NETLIB LU and condition number
> C     juergen v.Hagen 1999
>       integer info
>       real*8 anorm
>       include "NEC2DPAR.INC"
>       complex*16 work(2*MAXSEG)
>       real*8 rwork(2*MAXSEG)
>
>       integer msolver, compcn
>       common /msolverpar/ msolver, compcn
>       save /msolverpar/
> C***
>
3663c3695,3712
< 1     CALL FACTR (NP,A(KA),IP(KA),NROW)
---
>       if (msolver .eq. 0) then
>       CALL FACTR (NP,A(KA),IP(KA),NROW)
> C     netlib LU
>       else if (msolver .eq. 1) then
> C     compute condition number
>         if (compcn .eq. 1) then
>            anorm = zlange("1", NP, NP, A(KA), nrow, work)
>         endif
>         call zgetrf (np, np, A(KA), nrow, IP(KA), info)
>         if (info .ne. 0) print *,'ZGETRF info=',info
> C     condition number second part
>         if (compcn .eq. 1) then
>            call zgecon("1", NP, A(KA), NROW, anorm, condnum,
>      &                 work, rwork, info)
>            WRITE(3,'(20X,"CONDITION NUMBER ",G)') 1.0d0/condnum
>         endif
>       endif
> 1     continue
8336c8385
<       DIMENSION A(1), B(N1C,1), C(N1C,1), D(N2CZ,1), IP(1), XY(1)
---
>       DIMENSION A(*), B(N1C,*), C(N1C,*), D(N2CZ,*), IP(*), XY(*)
8507c8556,8561
<       DIMENSION A(1), IP(1), B(NEQ,NRH)
---
>       DIMENSION A(*), IP(*), B(NEQ,NRH)
>       integer info
>       integer msolver, compcn
>       common /msolverpar/ msolver, compcn
>       save /msolverpar/
>
8567c8621,8626
< 14    CALL SOLVE (NPEQ,A(IB),IP(IA),B(IA,IC),NROW)
---
>         if (msolver .eq. 0) then
>           CALL SOLVE (NPEQ,A(IB),IP(IA),B(IA,IC),NROW)
>         else if (msolver .eq. 1) then
>           call zgetrs("T",NPEQ,NRH,A(IB),nrow,IP(IA),B(IA,IC),NROW,info)
>         end if
> 14    continue
```


(b) Changes to the nec2d code for using the optimized routines - changes to subroutine `SOLVES`.

Fig. 1. Changes to `FACTRS` and `SOLVES`

.

```
28a29,31
>       INTEGER msolver, compcn
>       COMMON /MSOLVERPAR/ msolver, compcn
>       SAVE /MSOLVERPAR/
67c70
<       DIMENSION ATST(22),PNET(6),HPOL(3),IX(2*MAXSEG)
---
>       DIMENSION ATST(23),PNET(6),HPOL(3),IX(2*MAXSEG)
77c80
<      1 'NX','EN','TL','PT','KH','NH','PQ','EK','WG','CP','PL'/
---
>      1 'NX','EN','TL','PT','KH','NH','PQ','EK','WG','CP','PL','MS'/
85a89,91
>       write(*,*) ' NEC 2 - LAPACK version '
>       write(*,"('compiled for ',I5,' segments')") MAXSEG
>       write(*,"('maximum matrix size ',I5)") MAXMAT
177a184,186
>       msolver = 1
>       compcn = 0
> C***
230a240
>       IF (AIN.EQ.ATST(23)) GO TO 331
423a434,442
> C***
> C
> C     SOLVER FLAGS
> C
> 331   continue
>       if (ITMP1 .EQ. 0) msolver = 0
>       if ((ITMP1 .NE. 0) .AND. (ITMP2 .NE. 0)) compcn = 1
> C***
>       GO TO 14
3658c3677,3690
<       DIMENSION A(1), IP(NROW), IX(NROW)
---
>       DIMENSION A(*), IP(NROW), IX(NROW)
> C     additional variables for NETLIB LU and condition number
> C     juergen v.Hagen 1999
>       integer info
>       real*8 anorm
>       include "NEC2DPAR.INC"
>       complex*16 work(2*MAXSEG)
>       real*8 rwork(2*MAXSEG)
>
>       integer msolver, compcn
>       common /msolverpar/ msolver, compcn
>       save /msolverpar/
> C***
>
3663c3695,3712
< 1     CALL FACTR (NP,A(KA),IP(KA),NROW)
---
>       if (msolver .eq. 0) then
>       CALL FACTR (NP,A(KA),IP(KA),NROW)
> C     netlib LU
>       else if (msolver .eq. 1) then
> C     compute condition number
>         if (compcn .eq. 1) then
>           anorm = zlange("1", NP, NP, A(KA), nrow, work)
>         endif
>         call zgetrf (np, np, A(KA), nrow, IP(KA), info)
>         if (info .ne. 0) print *,'ZGETRF info=',info
> C     condition number second part
>         if (compcn .eq. 1) then
>           call zgecon("1", NP, A(KA), NROW, anorm, condnum,
>      &                work, rwork, info)
>           WRITE(3,'(20X,"CONDITION NUMBER ",G)') 1.0d0/condnum
>         endif
>       endif
> 1     continue
8336c8385
<       DIMENSION A(1), B(N1C,1), C(N1C,1), D(N2CZ,1), IP(1), XY(1)
---
>       DIMENSION A(*), B(N1C,*), C(N1C,*), D(N2CZ,*), IP(*), XY(*)
8507c8556,8561
<       DIMENSION A(1), IP(1), B(NEQ,NRH)
---
>       DIMENSION A(*), IP(*), B(NEQ,NRH)
>       integer info
>       integer msolver, compcn
>       common /msolverpar/ msolver, compcn
>       save /msolverpar/
>
8567c8621,8626
< 14    CALL SOLVE (NPEQ,A(IB),IP(IA),B(IA,IC),NROW)
---
>         if (msolver .eq. 0) then
>           CALL SOLVE (NPEQ,A(IB),IP(IA),B(IA,IC),NROW)
>         else if (msolver .eq. 1) then
>           call zgetrs("T",NPEQ,NRH,A(IB),nrow,IP(IA),B(IA,IC),NROW,info)
>         end if
> 14    continue
```

(a) Changes to the nec2d code for using the optimized routines - changes to use the new `MS`-card.

```
1906c1925
<       CALL HSFLD (XI,YI,ZI,0.)
---
>       CALL HSFLD (XI,YI,ZI,0.0d0)
3012c3031
<       CALL SFLDS (0.,EGND)
---
>       CALL SFLDS (0.0d0,EGND)
3175c3194
<       TYPE 1,MSG(IND+2:MSGLEN)
---
> C      TYPE 1,MSG(IND+2:MSGLEN)
3423,3424c3442,3443
<       DIMENSION A(1), B(N1C,1), C(N1C,1), D(N2C,1), BX(N1C,1), IP(1), IX
<      1(1)
---
>       DIMENSION A(*), B(N1C,*), C(N1C,*), D(N2C,*), BX(N1C,*), IP(*), IX
>      1(*)
4934c4983
<       CALL TEST (T01R,T10R,TE1R,T01I,T10I,TE1I,0.)
---
>       CALL TEST (T01R,T10R,TE1R,T01I,T10I,TE1I,0.0d0)
4947c4996
<       CALL TEST (T11R,T20R,TE2R,T11I,T20I,TE2I,0.)
---
>       CALL TEST (T11R,T20R,TE2R,T11I,T20I,TE2I,0.0d0)
5442c5491
<       CALL TEST (T01R,T10R,TE1R,T01I,T10I,TE1I,0.)
---
>       CALL TEST (T01R,T10R,TE1R,T01I,T10I,TE1I,0.0d0)
5458c5507
<       CALL TEST (T11R,T20R,TE2R,T11I,T20I,TE2I,0.)
---
>       CALL TEST (T11R,T20R,TE2R,T11I,T20I,TE2I,0.0d0)
5768c5817
<      1),ZLC(ISTEP),0.,0.,0.,' SERIES ')
---
>      1),ZLC(ISTEP),0.0d0,0.0d0,0.0d0,' SERIES ')
5771c5820
<      1),ZLC(ISTEP),0.,0.,0.,'PARALLEL')
---
>      1),ZLC(ISTEP),0.0d0,0.0d0,0.0d0,'PARALLEL')
5774c5823
<      1),ZLC(ISTEP),0.,0.,0.,' SERIES (PER METER) ')
---
>      1),ZLC(ISTEP),0.0d0,0.0d0,0.0d0,' SERIES (PER METER) ')
5777c5826
<      1),ZLC(ISTEP),0.,0.,0.,'PARALLEL (PER METER)')
---
>      1),ZLC(ISTEP),0.0d0,0.0d0,0.0d0,'PARALLEL (PER METER)')
5779,5780c5828,5829
<  23   CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),0.,0.,0.,ZLR(ISTEP),
<      1ZLI(ISTEP),0.,'FIXED IMPEDANCE ')
---
>  23   CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),0.0d0,0.0d0,0.0d0,
>      1ZLR(ISTEP),ZLI(ISTEP),0.0d0,'FIXED IMPEDANCE ')
5782,5783c5831,5832
<  24   CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),0.,0.,0.,0.,0.,ZLR(I
<      1STEP),'  WIRE  ')
---
>  24   CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),0.0d0,0.0d0,0.0d0,
>      10.0d0,0.0d0,ZLR(ISTEP),'  WIRE  ')
7150c7199
<       CALL HSFLD (XI,YI,ZI,0.)
---
>       CALL HSFLD (XI,YI,ZI,0.0d0)
7989c8038
<       CALL TEST(TMAG1,TMAG2,TR,0.,0.,TI,DMIN)
---
>       CALL TEST(TMAG1,TMAG2,TR,0.0d0,0.0d0,TI,DMIN)
8016c8065
<       CALL TEST(TMAG1,TMAG2,TR,0.,0.,TI,DMIN)
---
>       CALL TEST(TMAG1,TMAG2,TR,0.0d0,0.0d0,TI,DMIN)
8336c8385
<       DIMENSION A(1), B(N1C,1), C(N1C,1), D(N2CZ,1), IP(1), XY(1)
---
>       DIMENSION A(*), B(N1C,*), C(N1C,*), D(N2CZ,*), IP(*), XY(*)
```

(b) Changes for double precision.

Fig. 2. Other minor changes.

TABLE I

Optimized mathematical subroutine libraries.

Platform	Library and Supplier
i86 - Windows	Math Kernel Library by Intel `http://developer.intel.com/software/products/mkl/index.htm`
i86 - Linux	ASCI-RED by University of Tennessee in Kentucky, UTK `http://www.cs.utk.edu/~ghenry/distrib/archive.htm`
PPC - AIX	ESSL by IBM `http://www.research.ibm.com/mathsci/ams/ams_ESSL.htm`
Sparc - Solaris	sunperf by SUN `http://www.sun.com/`
PA - HP	MLIB by HP `http://www.hp.com/`
Athlon - Linux	Atlas, distributed e.g. by netlib `http://math-atlas.sourceforge.net/`
Any - Any	Atlas, distributed e.g. by netlib `http://math-atlas.sourceforge.net/`

TABLE II

Run Times for 2096 unknowns, original and optimized LAPACK factorization routines

Platform	Filling	Original Time	LAPACK
Athlon 900 MHz	6.040 s	68.990 s	4.740 s
PA 8200, 240 MHz	18.630 s	237.710 s	6.500 s
PA 8600, 552 MHz	7.120 s	148.200 s	2.810 s

[2] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. D. Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide*. Philadelphia, PA, USA: SIAM, 3 ed., 1999.

[3] J. J. Dongarra, J. du Croz, S. Hammarling, and R. J. Hanson, "An extended set of fortran basic linear algebra subprograms," *ACM Transactions on Mathematical Software*, vol. 14, pp. 1–17, Mar. 1988.

[4] J. J. Dongarra, J. du Croz, S. Hammarling, and I. Duff, "A set of level 3 basic linear algebra subprograms," *ACM Transactions on Mathematical Software*, vol. 16, pp. 1–17, Mar. 1990.

[5] J. J. Dongarra, J. du Croz, S. Hammarling, and I. Duff, "Algorithm 679: A set of level 3 basic linear algebra subprograms: Model implementation and test programs," *ACM Transactions on Mathematical Software*, vol. 16, pp. 18–28, Mar. 1990.

[6] J. J. Dongarra and D. W. Walker, "Software libraries for linear algebra computations on high performance computers," *SIAM Review*, vol. 37, pp. 151–180, June 1995.

[7] B. Kågström, P. Ling, and C. van Loan, "GEMM-based level 3 BLAS: Portability and optimization issues," *ACM Transactions on Mathematical Software*, vol. 24, pp. 303–316, Sept. 1998.

[8] B. Kågström, P. Ling, and C. van Loan, "Algorithm 784: GEMM-based level 3 BLAS: High-performance model implementations and performance evaluation benchmark," *ACM Transactions on Mathematical Software*, vol. 24, pp. 268–302, Sept. 1998.

[9] M. J. Daydé and I. S. Duff, "The RISC BLAS: A blocked implementation of level 3 BLAS for RISC processors," *ACM Transactions on Mathematical Software*, vol. 25, p. 316340, Sept. 1999.

[10] S. Seny, S. Chatterjeez, and N. Dumirx, "Towards a theory of cache-efficient algorithms," *Eleventh ACM-SIAM Symposium on Discrete Algorithms 2000*, vol. 1, Oct. 2000.

[11] R. C. Whaley and J. Dongarra, "Automatically tuned linear algebra software," tech. rep., Knoxville, Tennessee, USA, 1998.

[12] R. C. Whaley, A. Petitet, and J. J. Dongarra, "Automated empirical optimization of software and the atlas project," *To appear in Parallel Computing*, 2001. Also avalable as University of Tennessee LAPACK Working Note #147, UT-CS-00-448, 2000 (www.netlib.org/lapack/lawns/lawn147.ps).