MODELER'S NOTES

Gerald J. Burke

On a continuing theme, we have an update on NEC–4 performance on P-4 systems, including a contribution from Larry Laitinen on a low cost clone P-4 system. But first an embarrassing "woops". The changes that I recommended in the July 2000 and March 2001 Newsletters turned out to have some adverse side effects. A hopefully better fix is given below.

This issue first came up when Mike Morgan at the Naval Postgraduate School encountered problems with extraneous spikes in the near magnetic field over a Sommerfeld ground. NEC–4 evaluates the magnetic field from finite differences of electric field values. Such "glitches" were a known problem at points where the electric field evaluation is switched from interpolation to least-squares approximations and to asymptotic approximations, since the differences magnify small discontinuities in E. The spikes are very localized, but can be a serious problem if you are looking for peak field values. At that time I looked at the code and found that the largest spike coincided with a switch to using single-point integration on the segments to save time when the interaction distance was sufficiently large. Disabling this switch, so that the code always integrated over the segments, eliminated the worst spike in H. What I did not notice is that this change also locked out the asymptotic approximation for E, so that the least-squares approximation was used outside of its intended range. The L.S. approximation does allow extrapolation, but the error grows with increasing distance, and is not acceptable beyond a few wavelengths. In April of this year John Grebenkemper reported inaccurate impedance values as a short vertical dipole passed through a height of $\lambda/2$. This problem also appeared to be fixed by disabling the switch to single-point integration.

Instructions for disabling the switch to single point integration were included in Modeler's Notes in the July 2000 and March 2001 Newsletters. If you have made those changes, undo them by reactivating the four lines of code. Fortunately this problem did not affect the ground wave obtained with the RP1 command, so that may limit the damage.

Fixing the problems is a little more complicated. The error in the impedance of a vertical dipole with height around $\lambda/2$ is probably the most important, since it affects near field interactions. The problem is due to the treatment of point charges on the segment ends. An isolated segment has point charges at its ends that contribute to the electric field. In NEC the basis functions ensure continuity of current over the structure, so the fields due to point charges would cancel and are not evaluated. For the Sommerfeld-ground field at distances less than the limits for integration NEC integrates numerically for $(E_{\text{Sommerfeld}} - E_{\text{quasistatic}})$ and adds the analytic integral over the current for $E_{\text{quasistatic}}$. The quasistatic term has the same form as the free space field, and the evaluation omits the point charges for each segment. When the code switches to single-point integration it evaluates only $E_{\text{Sommerfeld}}$ with point charges (for a hertzian dipole) included. Hence there is a mismatched charge at the transition point.

There are a couple of ways that this problem might be fixed. I chose to turn on the evaluation of point charges before switching to single-point integration. The switch must be done for all segments at a junction, so it is necessary to compute the location of the segment ends rather than just the center. The modification to subroutine EFLD is shown below. The bold text is added code, and the call to subroutine EFLDSG has been modified.

```
C
C    FOR SOMMERFELD GROUND EVALUATION, TEST IF DISTANCE IS GREAT ENOUGH
C    TO USE POINT SOURCE APPROXIMATION. IF SO THE TOTAL REFLECTED
```

```
C    FIELD IS EVALUATED BY SOMMERFELD-INTERPOLATION CODE.
C
     IF(IPERF.EQ.2)THEN
       IMAGF=1
       RHO=SQRT(XIJ*XIJ+YIJ*YIJ)*GSCAL
       ZIJS=ZIJ*GSCAL
       IF((ZIJS.GT.ZZMX1).OR.(-ZIJS.GT.ZPMX1).OR.(RHO.GT.RHMX1))THEN
         IMAGF=2
         GO TO 17
       END IF
C      Include point charges on segment ends within 0.1 of the boundary
C      for switching to one-point integration.
        SLENH=.5*SLENJ
        XDST1=XI-(XJ-DXJ*SLENH)
        YDST1=YI-(YJ-DYJ*SLENH)
        ZDST1=ZI+(ZJ-DZJ*SLENH)*GSCAL
        RHOD1=SQRT(XDST1**2+YDST1**2)*GSCAL
        XDST2=XI-(XJ+DXJ*SLENH)
        YDST2=YI-(YJ+DYJ*SLENH)
        ZDST2=ZI+(ZJ+DZJ*SLENH)*GSCAL
        RHOD2=SQRT(XDST2**2+YDST2**2)*GSCAL
        IF(ZDST1.GT.ZZMX1-.1.OR.-ZDST1.GT.ZPMX1-.1.OR.
     &     RHOD1.GT.RHMX1-.1)THEN
          INDX1=60000
        ELSE
          INDX1=IND1
        END IF
        IF(ZDST2.GT.ZZMX1-.1.OR.-ZDST2.GT.ZPMX1-.1.OR.
     &     RHOD2.GT.RHMX1-.1)THEN
          INDX2=60000
        ELSE
          INDX2=IND2
        END IF
      ELSE
        INDX1=IND1
        INDX2=IND2
      END IF
C
C     EVALUATE IMAGE FIELD
C
      CALL EFLDSG(XIJ,YIJ,ZIJ,DXJ,DYJ,DZJR,SLENJ,ARADJ,ZPEDS,XK,ETX,
     &XKSJ,INDX1,INDX2,TXK,TYK,TZK,TXS,TYS,TZS,TXC,TYC,TZC)
```

←Change IND1 to INDX1, IND2 to INDX2

Note that it is assumed above that the flag for evaluating the point charge is the value 60000. I have modified our codes to use 60000 for the point charge flag and 30000 to flag connection to a patch. Earlier copies of NEC–4 used 20000 and 10000, respectively, and people who have run into the limit of 10000 segments may have changed their codes to other values. In any case, set the correct flag value for INDX1 and INDX2 in the modified code.

The above correction fixed the problem that John Grebenkemper found in the impedance of a vertical dipole when the ends bracket $z = \lambda/2$. A similar error occurred in the electric

field of a segment when the horizontal distance was one wavelength, and that is also fixed by the above change. Fixing the point charge treatment removed some spikes in the near H, but others remained where the code switched from interpolation to least-squares approximation and to the asymptotic approximation. I fixed the spike on switching to the asymptotic approximation by locking the evaluation method. On the first evaluation of the six-point central difference approximation for $\nabla \times E$ the code determines, based on the source and evaluation point locations, whether to use interpolation, least-squares or asymptotic. For the latter two the modified code sets a flag to lock the method for the remaining five evaluations of E. The same could not be done easily for the switch from interpolation to least-squares, since the interpolation cannot be extrapolated by much, and there is already a smoothing applied at this transition.

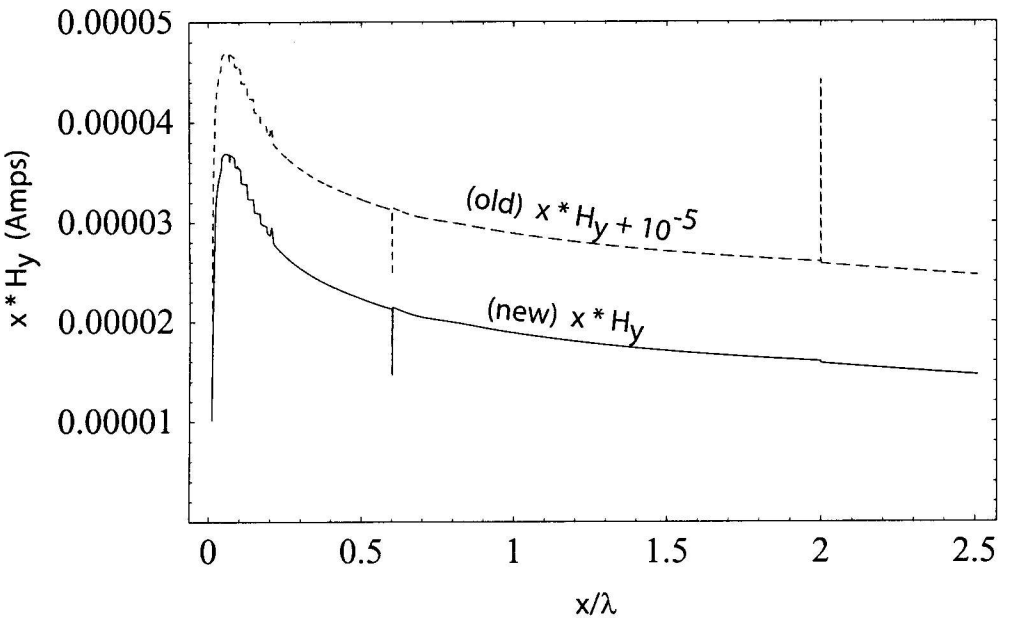

Fig. 1 Near magnetic field H_y times distance x due to a vertical 0.1λ dipole just above the ground. The complex ground permittivity was $\tilde{\epsilon}_r = 20 - j10$.

The near H fields before and after making these changes are shown in Figure 1 for a vertical 0.1λ dipole above a ground with complex permittivity of $\tilde{\epsilon}_r = 20 - j10$. The quantity $x \times H_y$ is plotted so that the singularity for small x does not dominate the plot. The remaining spike at about 0.6λ is due to the switch from interpolation to least squares. The curve for small x looks a little ragged. I have not looked for the source of this, but it is not noticeable if you plot H_y rather than $x \times H_y$. The best solution would be to write new code for interpolation and least-squares evaluation of H rather than numerically evaluating $\nabla \times E$. This would be reasonable to do now, with the advances in computer memory and speed since when NEC–4 was written. However, it would take a considerable amount of time to work out the details.

The changes to lock the evaluation method during the evaluation of $\nabla \times E$ are simple, but involve several areas of the code. I have not included them here to save space. Anyone who has NEC–4 and wants to make this change can contact me by email and I will send the changes, or can send new files for the code if they do not want to make the changes and recompile the code.

If anyone using NEC–4 would like a new code with these changes, they can let me know and I will send it out. I did get a call about eight to ten months ago from someone who reported that the field did not fall off correctly at large distances over ground, but I assumed that they were going so far that the near field evaluation was breaking down. It was probably a result of this problem, but I did not take a close enough look.

On a new topic, last May, just after the deadline for the last Newsletter, I got a Dell 8100 PC with 1.7 GHz P-4 processor and 1 GB of RAM for use at work. The cost at the "higher education" rate was just under $3000 with a 60 GB disk, CD-RW and DVD drives. Of course, a test of NEC running times was one of the first items of business, and the results were fairly impressive, as shown in the table below.

	CVF V. 6.5A compiler				CVF V. 6.6 compiler			
No. seg.	Fill (sec)	Factor (sec)	Total (sec)	Estimated MFLOPS	Fill (sec)	Factor (sec)	Total (sec)	Estimated MFLOPS
1200	5.49	7.80	13.90	590.8	6.21	7.36	14.23	626.2
2400	21.48	60.47	84.47	609.6	24.27	57.34	84.15	642.9
3600	48.39	202.29	256.28	615.0	54.81	192.13	252.55	647.5
7500	249.85	2413.04	2689.37	466.2	322.13	2135.46	2484.22	526.8

Times are given for NEC4D compiled with both the Compaq Visual Fortran V. 6.5A compiler and the new V. 6.6 compiler. I upgraded to V. 6.6 about three weeks ago, since V. 6.5A was generating bad code in compiling the EIGER program in debug mode. The reply from Compaq support was to get V. 6.6 from their ftp site as a free upgrade from V. 6.5A. At the time V. 6.6 was not mentioned on the Compaq website, but that may have changed by now. It is a main line, well documented release that adds some new features and fixes some bugs, including the one we encountered with EIGER. It also made some differences in the speed of NEC-4, as shown in the above table. The fill time went up and the factor time went down. The V. 6.6 compiler has a setting to optimize for a P-4 processor, but I could find no significant difference in speed with that setting over the default to optimize for a blend of processors. NEC-4 was compiled for "Maximum optimization", which did produce faster code than the default of "Full optimization", but changing the loop unrolling from the default of 4 to 8 did not have a significant effect. The NEC4D code that Larry Laitinen used for the results in the article that follows this were compiled with V. 6.5A.

In the above table it is seen that the speed in factoring the matrix has increased by about 6x for a 2x increase in clock speed over the 850 MHz P-III in the data that Larry Laitinen supplied in the July Newsletter. The increase in matrix fill speed falls a little short of the clock speed factor. Compiling the code for 7500 segments produced the message "Image size of 905,990,144 exceeds the maximum of 268,435,456, image may not run". It did run, but a little slower than scaling by N^3 from the smaller problems. The single processor 1.7 GHz P-4 is still about 25% slower in running EIGER than an older DEC Alpha with dual 533 MHz processors, which shows the advantage of a 64-bit system. It will be interesting to see how much faster the code runs on the P-4 with the Intel optimized LAPACK routine. If it shows the same factor of improvement as with the P-III it would really be impressive.

A recent check of the Dell website showed a 2 GHz P-4 available with 1 GB or RAM and DVD and CD-RW drives and no monitor for $2535 at the "higher education" rate. Filling it up to 2 GB of RAM cost another $1200. Larry Laitinen discusses a still more economical option in the following article.

Finally, if anyone can contribute modeling-related material for future newsletters, they are encouraged to contact our editor Bruce Archambeault or Jerry Burke, Lawrence Livermore National Lab., P.O. Box 808, L-154, Livermore, CA 94550, phone: 925-422-8414, FAX: 925-423-3144, e-mail: burke2@llnl.gov.