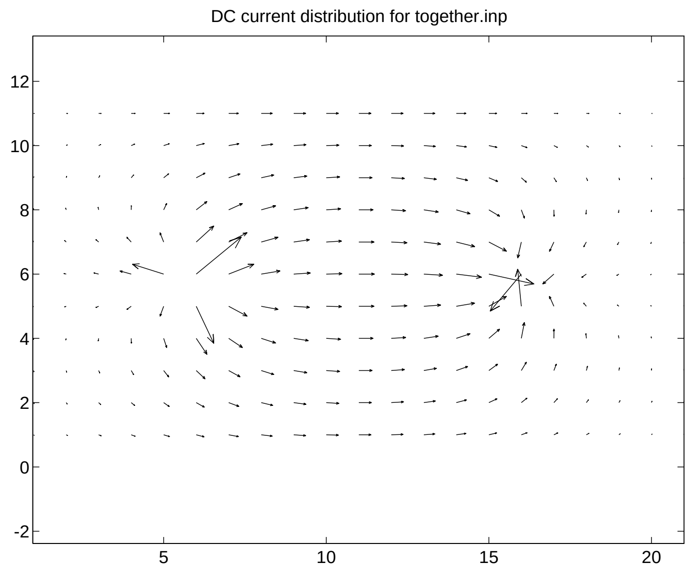

Figure 15:

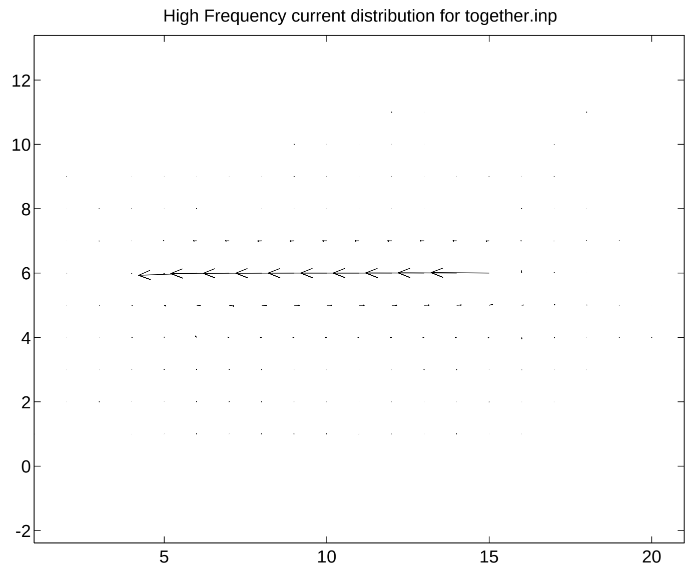

Figure 16:

```
                                Version 4.1
                                Jun 15 1993

Commands to get started: intro, demo, help help
Commands for more information: help, whatsnew, info, subscribe

>> load Grid1_0.mat
>> who

Your variables are:

grid1g1   grid2g1

>> quiver(real(grid1g1),real(grid2g1),2)
>> title('DC current distribution for together.inp')
>> print together_DC.ps
>> load Grid1_1.mat
>> who

Your variables are:

grid1g1   grid2g1

>> quiver(imag(grid1g1),imag(grid2g1),2)
>> title('High Frequency current distribution for together.inp')
>> print together_highf.ps
```

Note that for the DC case in Figure 15 the current spreads across the plane as it travels from its source to sink points but at high frequency the current is focused underneath the trace. Note also that the high frequency current travels in the opposite direction as the current at DC since $I = V/(jwL) = -jV/(wL)$.

4.2.2 Traces over a divided plane

Next consider the example file `broken.inp` shown in Figure 17. This is identical to `together.inp` except that the plane is now broken in two pieces and connected by copper "tethers" as shown.

After running FastHenry with

```
fasthenry broken.inp -d grids -x trace1
```

the two matrices can be concatenated in matlab to produce Figures 18 and 19 with the following commands:

```
>> load Grid1_0
>> who
```

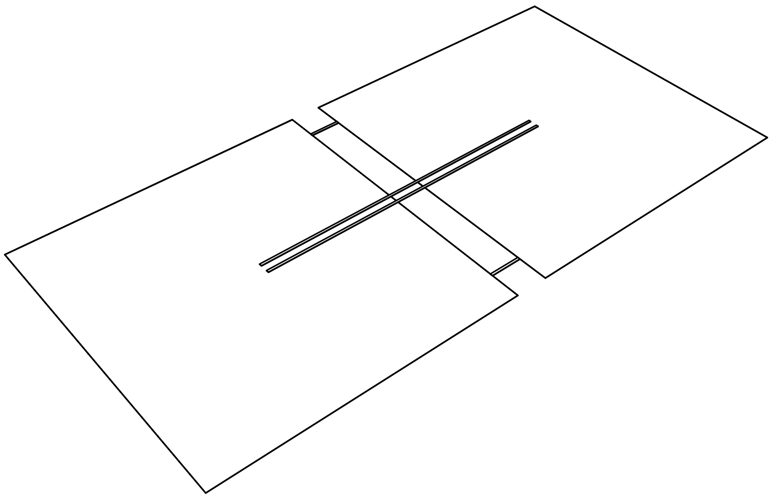

Figure 17: Two traces over a divided ground plane

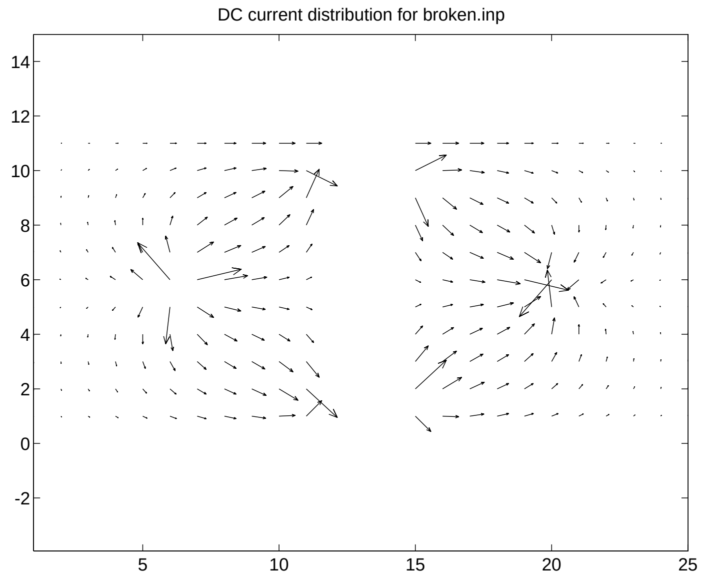

Figure 18:

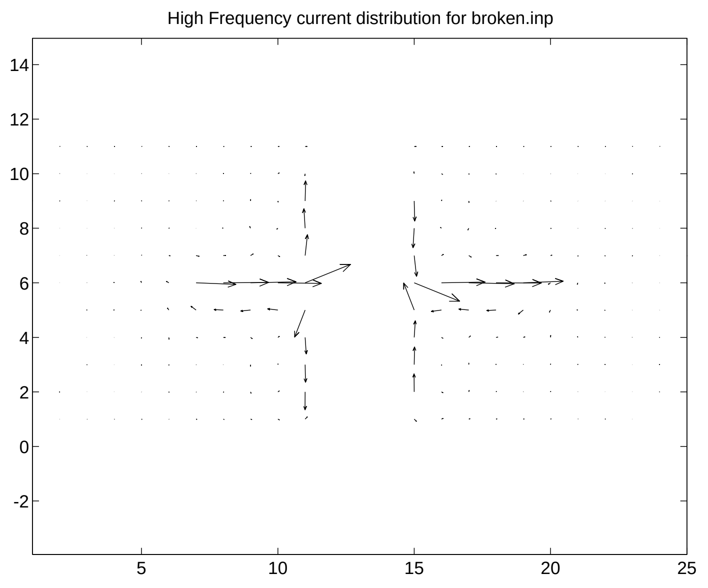

Figure 19:

```
Your variables are:

grid1g_one     grid2g_one
grid1g_two     grid2g_two

>> size(grid1g_one)

ans =

    11    11

>> space = zeros(11,3);
>> new1 = [-grid1g_two(:,11:-1:1) space grid1g_one];
>> new2 = [grid2g_two(:,11:-1:1) space grid2g_one];
>> quiver(real(new1),real(new2),2);

(similarly for Grid1_1.mat)
```

Note that some difficulty is involved in combining the matrices because the `seg1` direction in space is different for the two planes since the corners of the plane are defined in counter-clockwise order for one plane, and clockwise for the other.

From Figure 18 it may seem unusual that current can point into the empty space between the planes. This, however, is only an artifact of the method of visualization since the current vectors do not precisely represent the current flow at a particular point. Since the ground plane is discretized as a grid of segments (see Figure 4), one could attempt to represent the current as that passing through each node. But each node

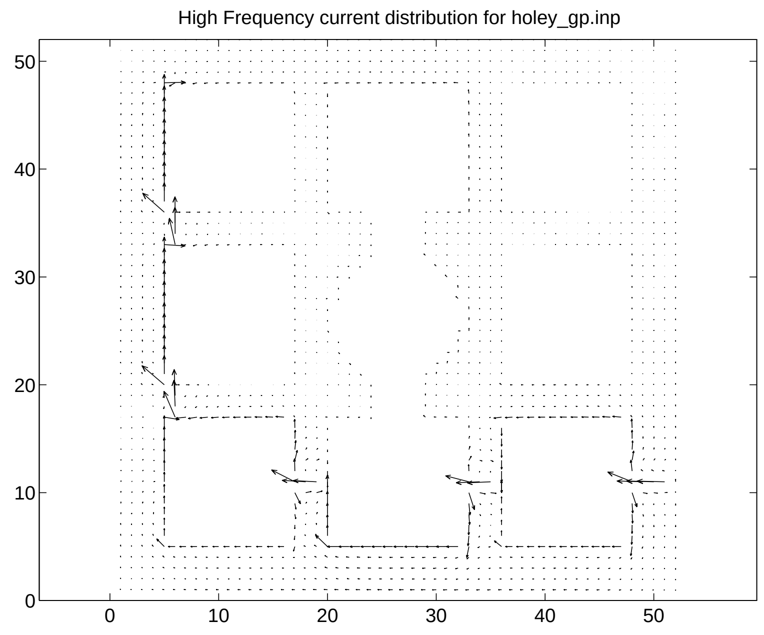

Figure 20:

on the plane has four segments attached to it while only two are needed to define a vector direction. FastHenry consistently picks the segments with the smaller indices. This choice can be altered in the function `makegrids()` in `fillM.c` if the user wishes to average the currents, for instance.

4.2.3 Trace over a plane with holes

As a final, more interesting example, consider observing the current for example file `holey_gp.inp` shown previously in Figure 7. To produce Figure 20,

```
fasthenry holey_gp.inp -d grids
```

and then in Matlab,

```
>> load Grid1_1
>> quiver(imag(grid1g1),imag(grid2g1),3)
```

Again in this high frequency case, the current is observed to try its best to stay underneath the trace, outlining the holes it must circumvent.

A Compiling FastHenry

A tar file containing the source files for `fasthenry` and this guide may be obtained on tape by sending a written request to

Prof. Jacob White
Massachusetts Institute of Technology

Department of Electrical Engineering and Computer Science
Room 36-880
Cambridge, MA 02139 U.S.A.

This address may also be used for general correspondence regarding `fasthenry`, although electronic mail may be sent to `fasthenry-bug@rle-vlsi.mit.edu`, for bug reports, and to `fasthenry@rle-vlsi.mit.edu`, for questions or comments, if it is more convenient. Comments on FastHenry are encouraged as well as comments on unclear portions of this manual.

The tar file has the form

```
fasthenry-3.0-9Oct96.tar.Z
```

and yields a one level directory when untarred with the commands

```
uncompress fasthenry-3.0-9Oct96.tar.Z
tar xvf fasthenry-3.0-9Oct96.tar
```

It will create a directory tree underneath `fasthenry-3.0` which contains all the C source files underneath `src/`, this manual in `doc/`, and the example files in `examples/`.

The tar files may also be obtained via anonymous ftp to `rle-vlsi.mit.edu`. Use username `anonymous` with your email address as the password. The FastHenry tar files are contained in directory `pub/fasthenry`. This directory also contains various publications on multiplole accelerated inductance extraction. The most complete description is the compressed postscript file `ms_thesis.ps.Z`:

> Mattan Kamon, *Efficient Techniques for Inductance Extraction of Complex 3-D Geometries*, M.S. thesis, Massachusetts Institute of Technology, Cambridge, MA., February 1994.

`mtt.ps.Z` is a preprint of

> M. Kamon, M.J. Tsuk, and J. White, "FASTHENRY: A Multipole-Accelerated 3-D Inductance Extraction Program", *IEEE Transactions on Microwave Theory and Techniques*, Vol. 42, September 1994.

See the `README` file at the ftp site for update information on releases, manuals, publications.

A.1 Compilation Procedure

FastHenry is compiled by changing to the `fasthenry-3.0` directory, and typing

```
make all
```

to create the executables `fasthenry` and `zbuf` in the `bin/` directory.

If you are compiling on a DEC 5000, DEC 3000 (Alpha), SGI, or a system running System V (HP and Suns running Solaris, perhaps), other flags are required for compilation. In this case, before compilation as instructed above, give the command:

```
config <name>
```

where `<name>` is one of `dec`, `alpha`, or `sgi`, or `solaris` for compilation on a DEC 5000, DEC 3000 (Alpha), and Silicon Graphics workstation, respectively. Additional steps for compiling on an SGI are given in the file `README.sgi` in the `fasthenry-3.0` directory. `config sysV` is also provided for compilation on HP-UX, and other machines running System V, but has not been thoroughly tested. For more details on compilation, see the `fasthenry-3.0/README` file.

A.2 Producing this Guide

This guide is available in postscript as three separate files: `manual_001.ps, manual_002.ps`, and `manual_003.ps` in the `doc/` directory. `manual_001.ps` contains up through page 13, `manual_002.ps` is pages 14–31, and `manual_003.ps` is pages 32–36. Note that these files contain many detailed postscript images and may take significant time to print. The LaTeX version of the manual without the figures is also available.

Also, the nonuniform plane discretization manual is in the files `nonuniform_manual_1.ps` and `nonuniform_manual_2.ps`

B Changes in Version 3.0

- Specify a nonuniform discretization of a reference plane to capture small features in fewer elements.

- Two approaches for generating spice equivalent circuits are available:

  1. An equivalent circuit for a single frequency.

  2. A circuit which models the frequency dependent resistances and inductances through a reduced state-space representation

  Comments: Method 1 will not model frequency dependent resistance and inductance since it gives and R and L at the single specified frequency. Method 2 will model the full effects up to some frequency.

- Major Bug fix: For reference planes which form segments with different widths in the x-direction versus the y-direction, the sizes weren't computed correctly in version 2.0 and later. This has been fixed.

- The `zbuf` program now takes the "-m" argument to produce a Matlab file for faster visualization in matlab. This is very beneficial for large files since producing the postscript file can take n^2 time. The matlab file can be viewed within matlab with the fasthenry-3.0/bin/plotfastH.m matlab function. The file `zbuffile.mat` would be produced with "zbuf -m zbuffile" which can then be viewed in matlab with "`>> plotfastH('zbuffile.mat')`". Also, you can modify the file `src/zbuf/dump_struct.c` to output in YOUR own format instead of matlab.

- Sparse preconditioner. Specify `-p shells` to use a preconditioner based on current shell sparsification.
- Regurgitate the input file with `-v` to see what FastHenry thinks it has read. Also can translate and reflect geometry before output.