gnuplot

or, How To Make Your Data Look Neat and Shiny

Ron Ho
3/14/01
ronho@vlsi.stanford.edu

Introduction

- Plotting data into pretty charts is pretty standard fare
  - Ultimate consumer: journals, conference papers, thesis
  - Immediate consumers: Framemaker, Latex, (troff!?)

- Sources of data often can produce pretty plots themselves
  - Matlab, Mathematica, Mathcad

- But more often, we get raw data outside of nifty software
  - Lab measurements, simulations, C/Perl code
  - … or we're dissatisfied with other tool graphing capabilities

- How does one make data pretty and consumable?

gnuplot

- Tools have come a long way
  - magicplot.pl when I took ee371 and ee315 (a *long* time ago)
    - Took a text file and drew bar graphs in m1/m2/m3
    - Axes in poly, labels using wire lab on bits of diff

- I think the best plotting tool today is gnuplot
  - Very feature-rich
  - I am not an expert it, but I have learned a few tricks
  - I was going to cover matlab, too…
    - But decided I really didn't know matlab very well
    - Besides, this is a long talk already…

- Lots of demonstrations today
  - Which wreak havoc on creating useful slides, but we'll see

Introduction

- UNIX gnuplot 3.7.1 sits in /usr/pubsw/bin (AFS-land)
  - If you don’t mount AFS (why not?), you can compile it from
    - `ftp://ftp.gnuplot.org/pub/gnuplot`
  - Also available for win32 machines in precompiled format

- Offers 2D and 3D plotting with a wide variety of options
  - It has a pretty good online “help” feature: RTFM!

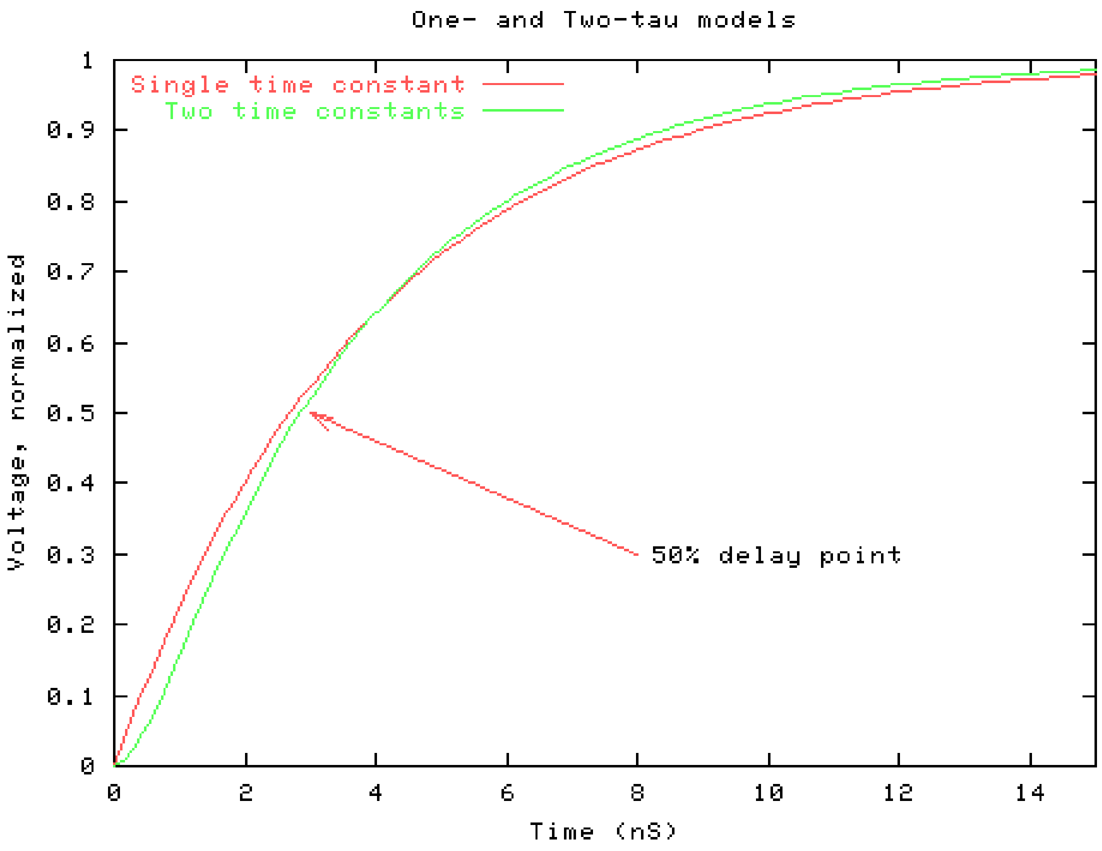
- gnuplot is, interestingly enough, not affiliated with FSF or GNU
  - Hence it’s called “gnuplot,” not “GNUplot”
    - Historical reason: authors wanted "newplot" but it was taken
  - Not GPL’ed, but plain old copyrighted freeware

1-My First Graph: basics

Script:

```
plot 1-exp(-x/3.8825)
pause –1
set xrange [0:15]; replot
plot 1-exp(-x/3.8825) title "Single time constant"
set xlabel "Time (nS)"; replot
set ylabel "Voltage, normalized"
set key top left
replot 1-(3.44*exp(-x/3.44)-0.44*exp(-x/0.44))/3.0 title "Two time constants"
set title "One- and Two-tau models"
set arrow 1 from 8,0.3 to 3.0,0.5 head
set label 1 "50% delay point" at 8.2,0.3 left
```

Each step is followed by a "replot"



Related commands:

```
set key x,y
set [no]log (x|y)
set autoscale (x|y)
```

- *Note: Screen shots are low-quality to keep the file size down. High-quality .eps plots discussed later.*

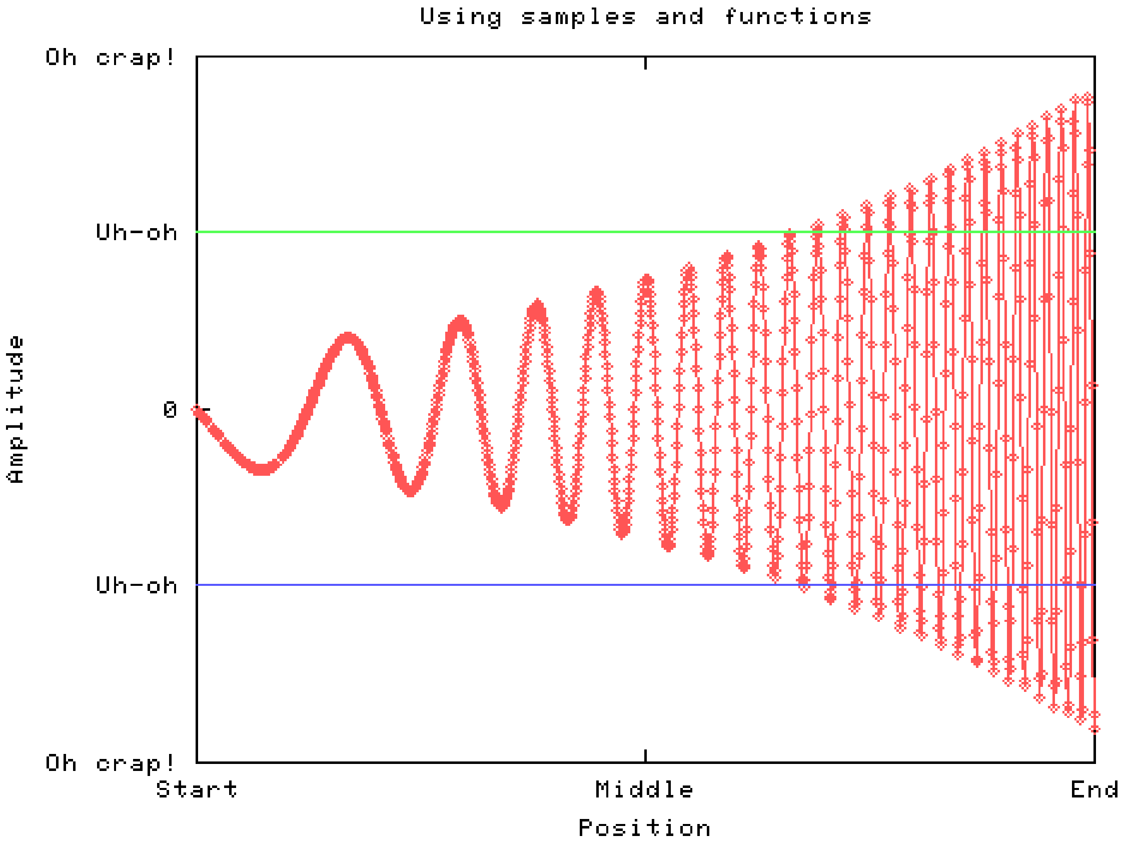
2-Plotting functions and sampling

Script:

```
clear; reset
set xrange [x1:x2]; set yrange [y1:y2]
set xlabel "…"; set ylabel "…"
set title "Using samples and functions"
f(x) = x**5
pi = 3.14159; sf = 4.5
plot (sf**x)*sin(f(x)*pi) notitle with linespoints
set samples 1000   ← normally, get 100 points
set xtics ("Start" 1, "Middle" 1.6, "End" 2.2)
set ytics ("Oh crap!" -30, "Uh-oh" -15, "0" 0, "Uh-oh" 15, "Oh crap!" 30)
replot 15 notitle; replot -15 notitle
```

Related commands:

```
show variables
show functions
high=110; f2c(t)=(x-32)*5.0/9.0
set yrange [f2c(20):f2c(high)]
```



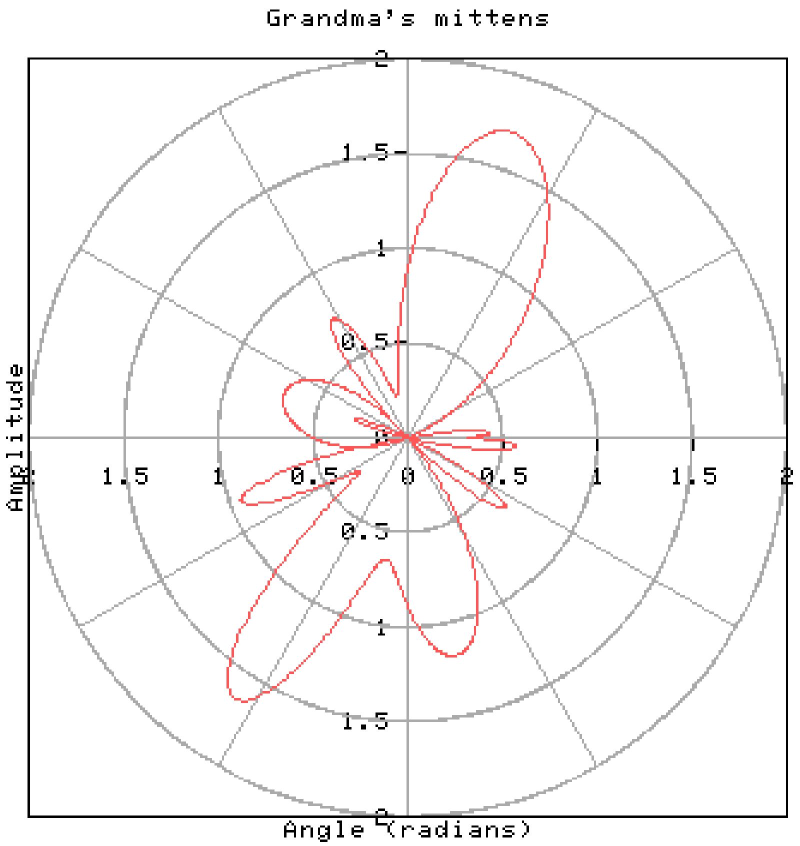
3-More 2D plots

Script:

```
➢set samples 1000
➢set xlabel; set ylabel; set title
➢set xrange [-pi:pi]   ← pi predefined!
➢plot sin(x)*cos(x) + sin(x)*sin(x) -
0.5*cos(2*x*x) notitle
➢set grid
➢set polar
➢set trange[-pi:pi]
➢plot sin(t)*cos(t) + sin(t)*sin(t) -
0.5*cos(2*t*t) notitle
➢set grid polar
➢set xtics axis; set ytics axis
➢set xrange [-2:2]; set yrange [-2:2]
➢set size square
➢set title "Grandma's mittens"
```

Related commands:

```
➢set size ratio aspectratio
➢set size xscale,yscale
➢set parametric  <- polar is a special case
```

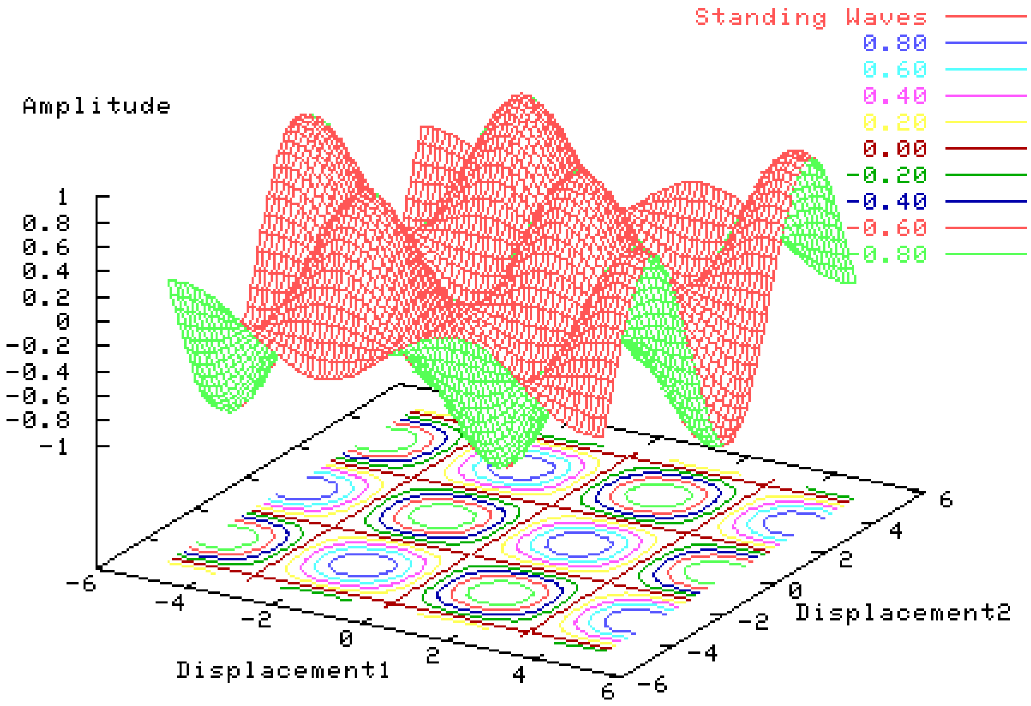


4-Basic 3D plots

Script:

- `set xlabel; set ylabel`
- `set zlabel "Amplitude"`
- `set parametric`
- `splot u,v,sin(u)*cos(v) title "Standing Waves"`
- `set isosamples 75,75` ← *10 is normal*
- `set contour base`
- `set cntrparam level incremental -1, 0.2, 10` ← *start,incr,num*
- `set clabel '%4.2f'` ← *C's scanf*
- `set contour surface`
- `set contour base; set nosurface`
- `set surface;`
- `set view 20,60`
- `set view 60,30` ← *xrot, zrot*
- `set hidden3d`

Just to illustrate "set parametric"; could also use `splot sin(x)*cos(y)` *w/o parametric*



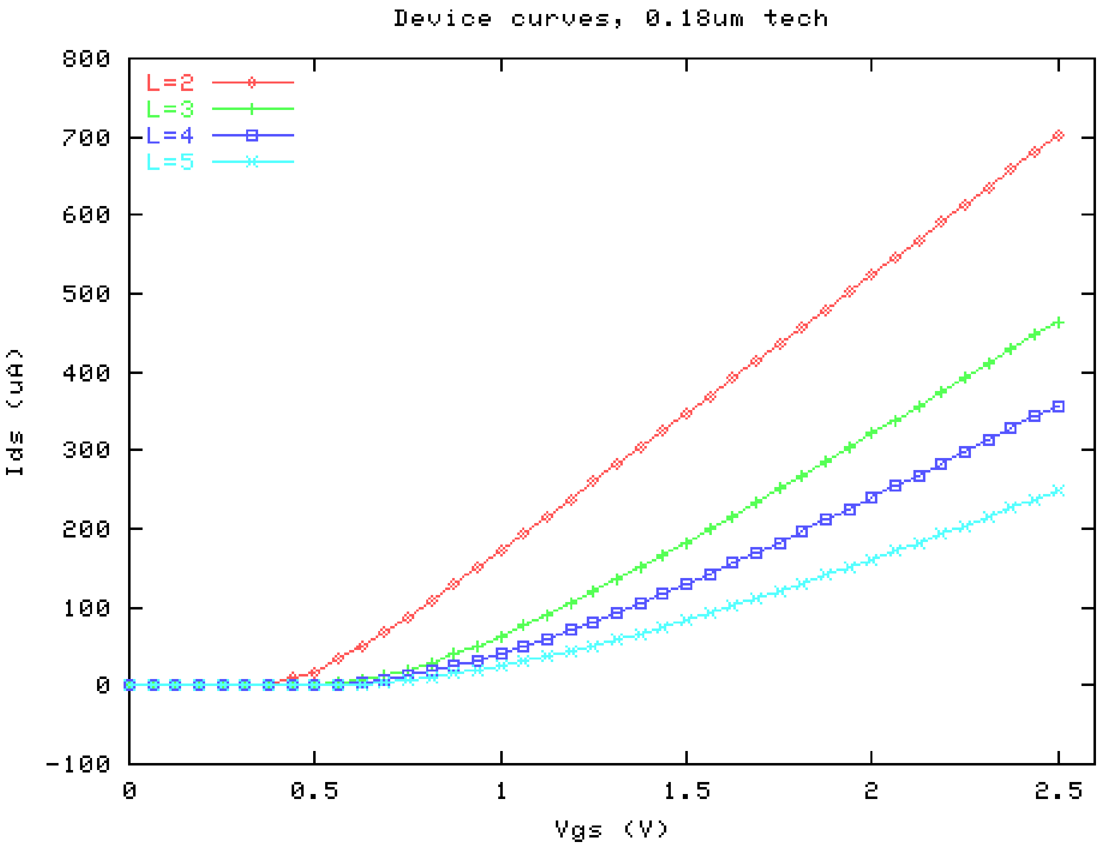
Related commands:

- `set contour [base|surface|both]`
- `set [no]surface`

5-Plotting from data files

Script:

```
➢set xlabel; set ylabel; set title
➢set key top left
➢plot "plot5.dat" title "IV curves"
➢plot "plot5.dat" using ($1*2.5/2e-
9):($2*-1e6) title "IV curves"
➢set xlabel "Vgs (V)"; set ylabel "…"
➢set xrange [0:2.6]
➢plot "plot5.dat" index 2 using
($1*2.5/2e-9):($2*-1e6) title "L=4"
➢replot "plot5.dat" index 3 using
($1*2.5/2e-9):($2*-1e6) title "L=5"
with lines
➢set data style linespoints
➢plot "plot5.dat2" u ($1*2.5/2e9):($3*-
1e6) title "L=3"
➢plot "plot5.dat3" u ($1*2.5/2e9):($2*-
1e6) '%lf,%lf,%lf,%lf,%lf' title "L=2"
```



Notes:

```
➢plot <FILE> index n …
```

requires \n\n between datasets

6-Plotting from data files

Script:

```
➢set xlabel; set ylabel; set title
➢set xrange [0:110]
➢plot "plot6.dat" u 1:3 t "Leakage" w p
```

only need "$" for expressions

```
➢plot "plot6.dat" u 1:3 t "Leakage" w l
➢plot "plot6.dat2" u 1:3 t "Leakage" w l
```

\n in data prevents line-connecting

```
➢set xrange [0.8:1.9]
➢set xlabel "Voltage (V)"
➢plot "plot6.dat2" u 2:3 t "Leakage" w l
➢set xrange [0:110]; set yrange [0.8:1.9]
➢set xlabel "Temperature (C)" ,-1
➢set ylabel "Voltage (V)" ,-1
```

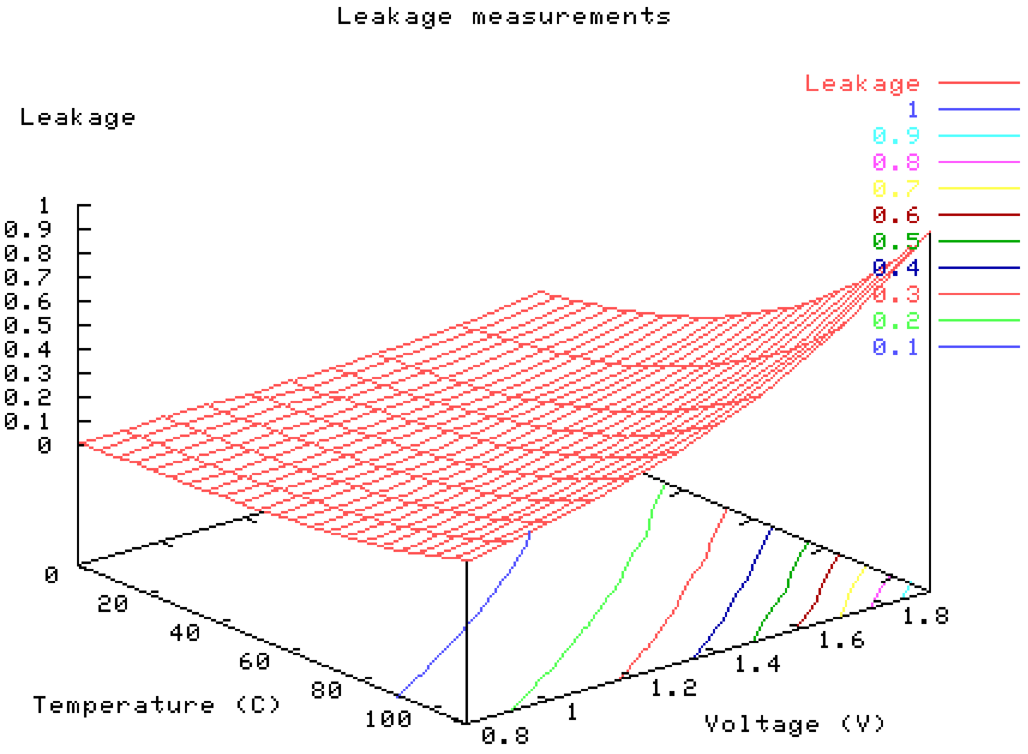
xoff=0,yoff=-1 in x's

```
➢set zlabel "Leakage"
➢splot "plot6.dat2" u 1:2:3 t "Leak" w l
```

splot using x:y:z

```
➢set view ,50
➢set contour base
➢set hidden3d
```

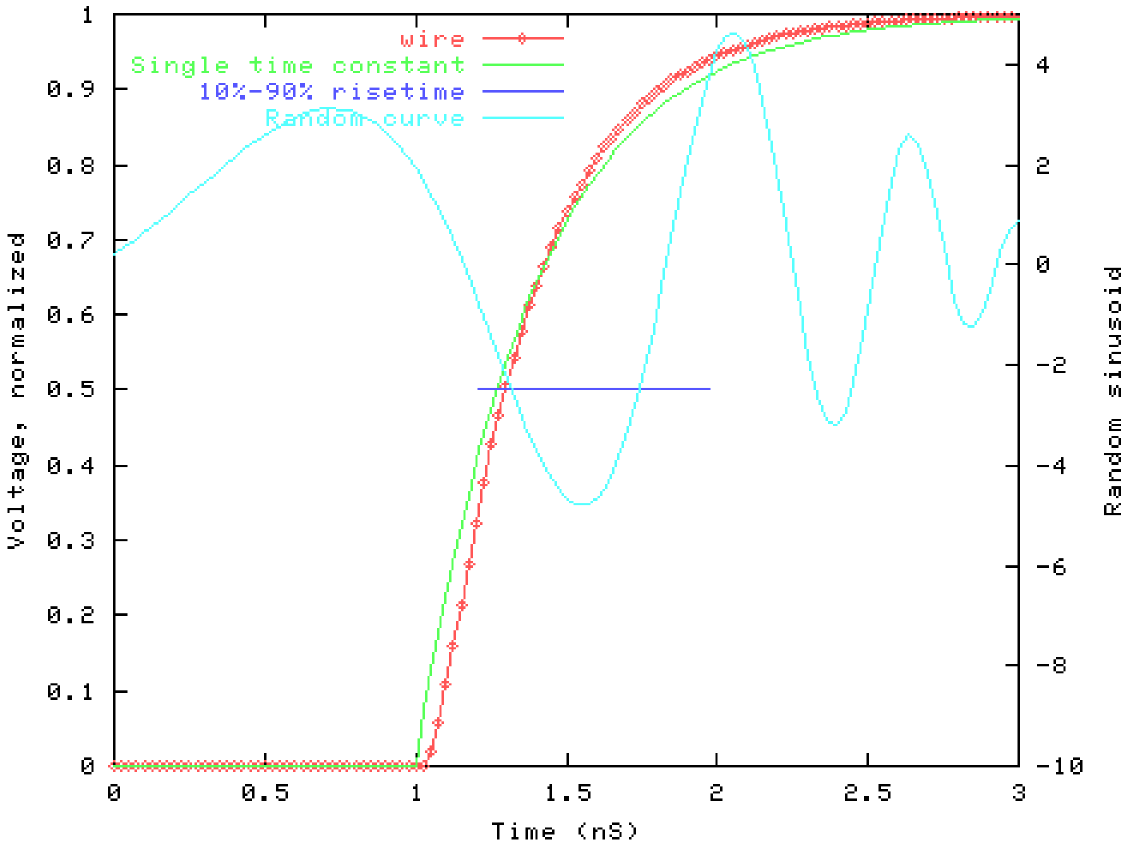
only works for lines or linespoints



7-Axes. Ternary operations.

Script:

```
➢set xrange; set xlabel; set ylabel
➢set key top left
➢plot "plot7.dat" u ($1*1e9):($2/1.8) t "wire" w lp
➢replot 1-exp(-(x-1)/.38825) t "Single time constant"
➢plot "plot7.dat" u ($1*1e9):($2/1.8) t "wire" w lp
➢replot (x<1) ? 0 : 1-exp(-(x-1)/.38825) t "Single time constant"
➢replot x>1.2 && x<2 ? 0.5:1/0 t "10%-90% risetime"
➢replot 5*sin(exp(x))*sin(x)+0.2 axes x1y2 t "Random curve"
➢set y2tics
➢set ytics nomirror
➢set y2label "Random sinusoid"
➢set y2range [-10:5]
```



Related commands:

```
➢plot 'file' u 1:($4<0?1/0:($2+$3)/2)
```

plots average of $2,$3 only if $4>=0

8-Nifty side-note

Ternary operator surprisingly powerful!

How quickly does $\sqrt{6\sum_{i=1}^{\infty}\frac{1}{i^2}}$ **converge to pi?**

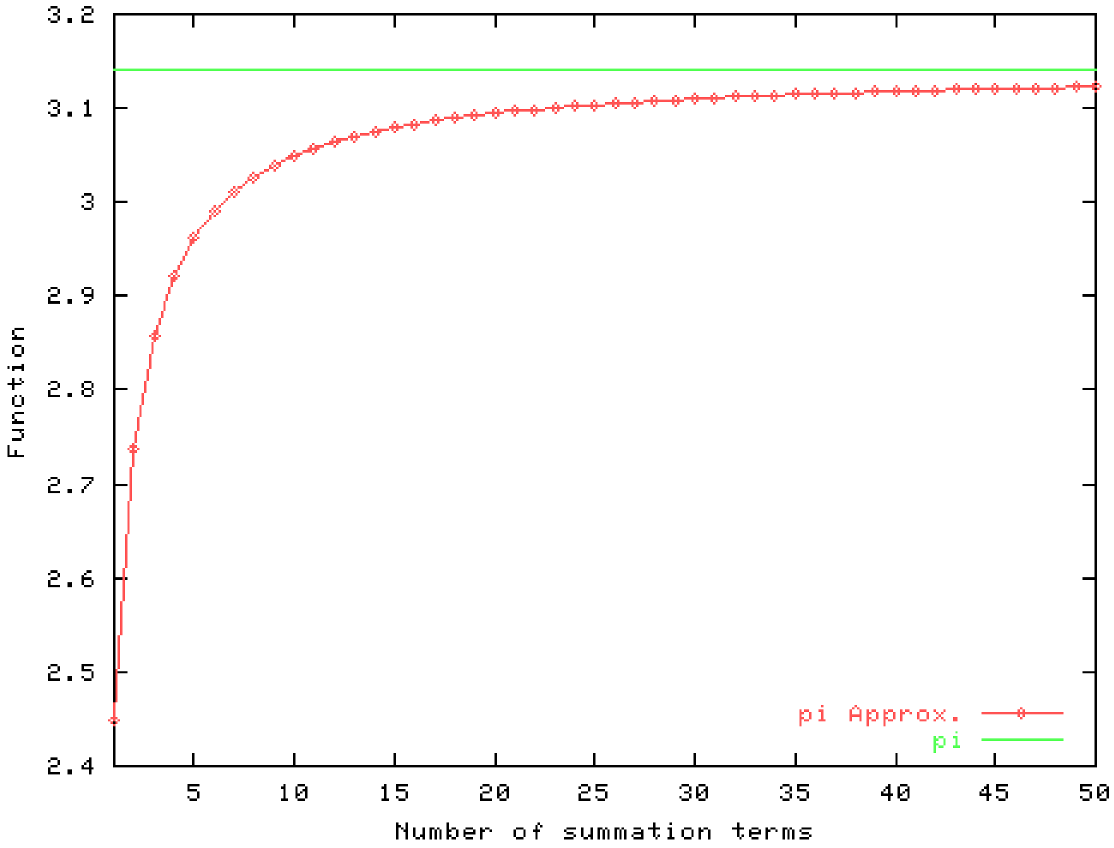
Script:

```
➢set xlabel "Number of summation terms"
➢set ylabel "Function"
➢set xrange [1:50]
➢set samples 50
```

← *critical: integers only!*

```
➢set key bottom right
➢f_part(x) = 1/(x*x)
➢f_sum(x) = f_part(x) + ((x>1) ?
f_sum(x-1) : 0)
➢f(x) = sqrt(6*f_sum(x))
➢plot f(x) title "pi Approx." w lp
➢replot pi
```

Answer: Not very quickly!



Note: *Stack space is limited; plotting from [0:100] runs out of stack space ☹ (do it using two functions)*

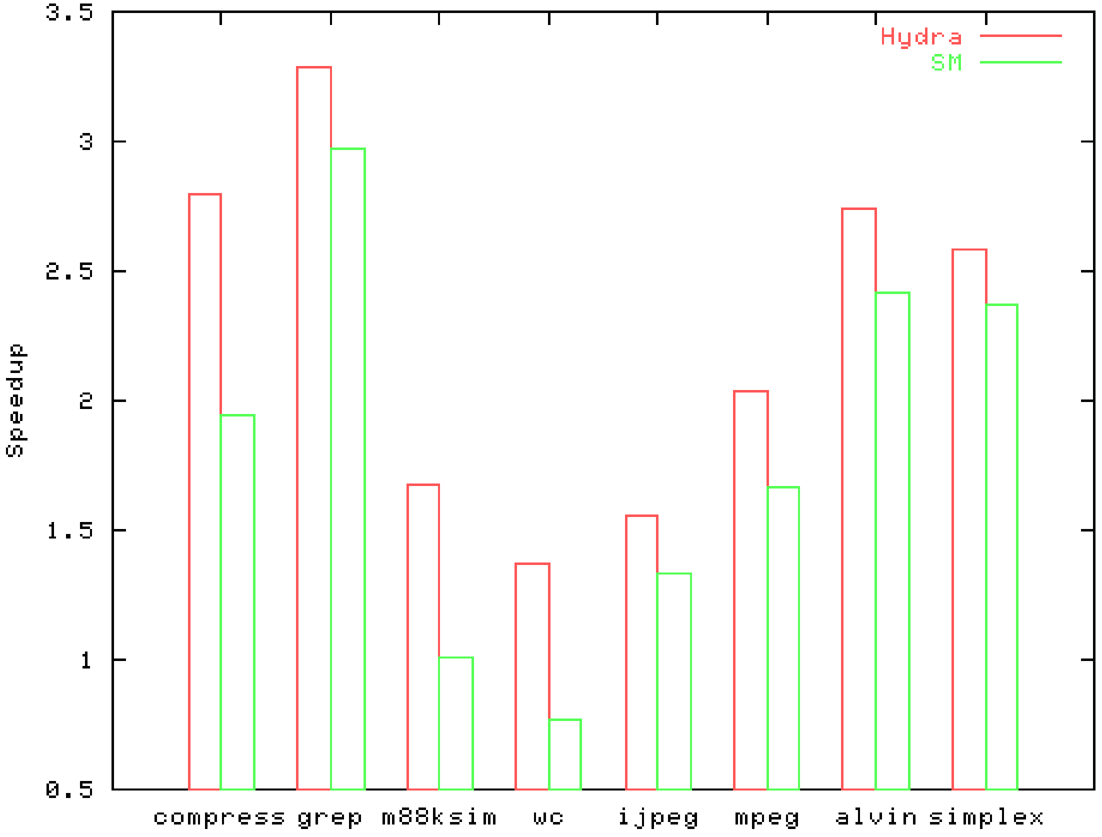
9-Bar graphs

Script:

```
➢set xtics ("compress" 1, "grep" 2,
"m88ksim" 3, "wc" 4, "ijpeg" 5, "mpeg"
6, "alvin" 7, "simplex" 8)
➢set ylabel "Speedup"
➢set xrange [0:9]
➢plot "plot8.dat" u 1:2 t "Hydra" w lp
➢replot "plot8.dat" u 1:3 t "SM" w lp
➢plot "plot8.dat" u 1:2 t "Hydra" w
boxes
➢replot "plot8.dat" u 1:3 t "SM" w
boxes
➢set boxwidth 0.3
➢plot "plot8.dat" u ($1-0.15):2 t
"Hydra" w boxes
➢replot "plot8.dat" u ($1+0.15):3 t
"SM" w boxes
```

Notes:

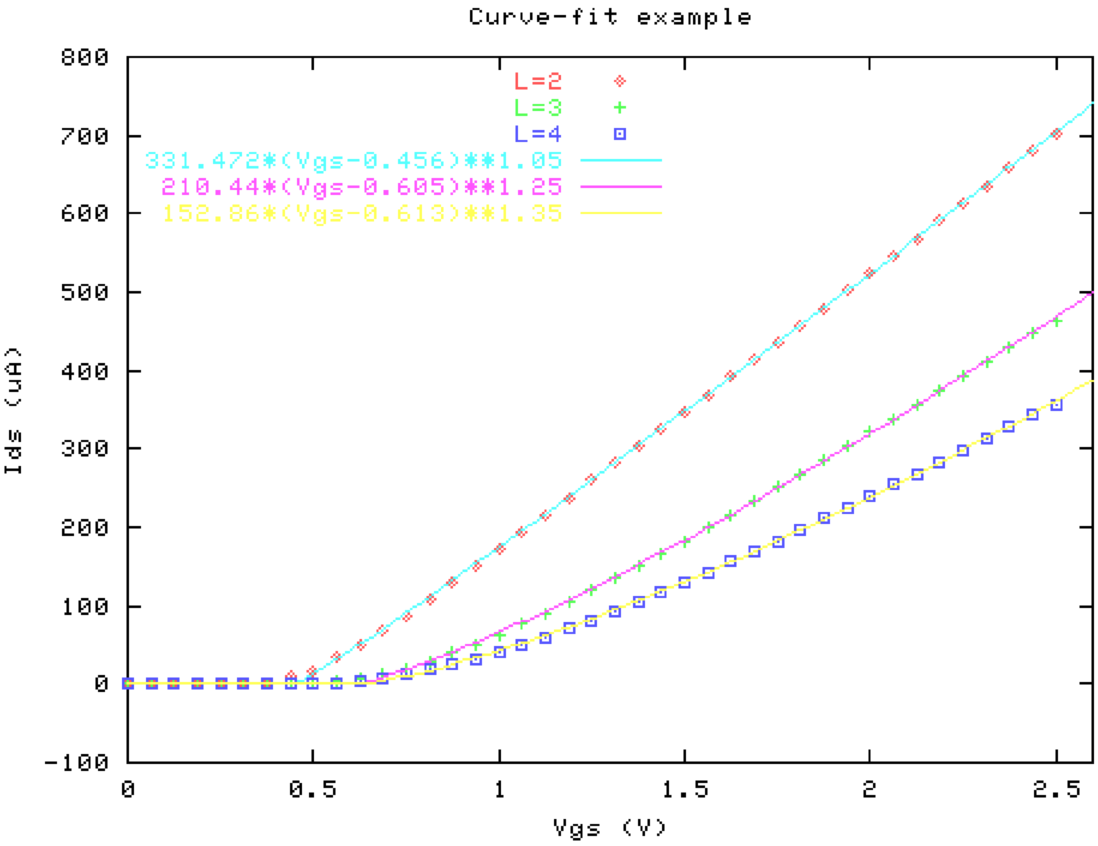
No way to fill in the boxes using stock gnuplot (although some post-processing hacks exist, including simply using Frame)



10-Curve-fitting

Script:

```
➢set xlabel; set ylabel; set title
➢set xrange [0:2.6]; set key
➢plot "plot5.dat2" u ($1*2.5/2e-9):($2*-1e6) t "L=2" w p
➢replot "plot5.dat2" u ($1*2.5/2e-9):($3*-1e6) t "L=3" w p
➢f1(x) = x>b1 ? a1*((x-b1)**c1) : 0
➢fit f1(x) "plot5.dat2" u ($1*2.5/2e-9):($2*-1e6) via a1,b1,c1
➢replot f1(x) title "331.472*(Vgs-0.456)**1.05" w l
➢f2(x) = x>b2 ? a2*((x-b2)**c2) : 0
➢fit f2(x) "plot5.dat2" u ($1*2.5/2e-9):($3*-1e6) via a2,b2,c2
➢replot f2(x) title "210.44*(Vgs-0.605)**1.25" w l
```



Notes:

Max 3000 data points for curvefitting

fit.log holds the iterative information

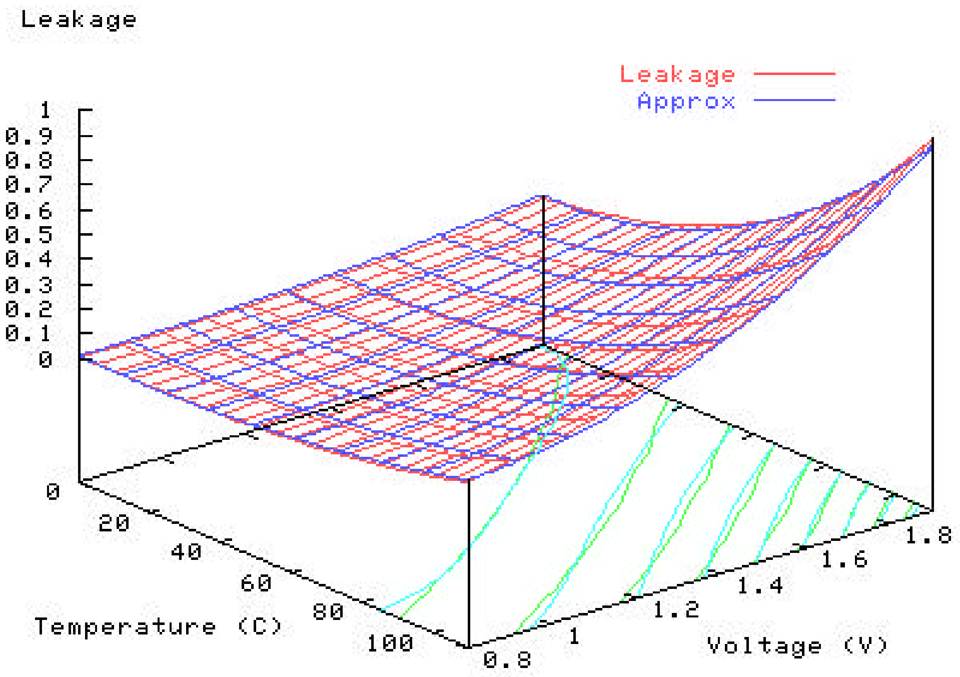
Must manually type in the fitted values for titles/labels. Most often requested feature for v3.8!

11-Curve-fitting, another example

Script:

```
➢(x|y}range; (x|y|z)label
➢set data style lines
➢set view ,50; set key 60,1.9,1
➢splot "plot6.dat2" u 1:2:3 t "Leakage"
➢f(x,y) = a+b*x+c*y
➢fit f(x,y) "plot6.dat2" u 1:2:3:(1) via
a,b,c
➢replot f(x,y)
➢splot "plot6.dat2" u 1:2:($3-f($1,$2)) not
➢f(x,y) = a+b*x+c*y*y+d*y+e*x*y*y+f*x*y
➢fit f(x,y) "plot6.dat2" u 1:2:3:(1) via
a,b,c,d,e,f; replot f(x,y)
➢splot "plot6.dat2" u 1:2:($3-f($1,$2)) not
➢f(x,y) = a + b*x*x + c*x + d*y*y + e*y +
f*x*x*y*y + g*x*x*y + h*x*y*y + i*x*y
➢fit f(x,y) "plot6.dat2" u 1:2:3:(1) via
a,b,c,d,e,f,g,h,i; replot f(x,y)
➢splot "plot6.dat2" u 1:2:($3-f($1,$2)) not
➢set contour base; set noclabel
➢splot "plot6.dat2" u 1:2:3 t "Leakage"
➢replot f(x,y) t "Approx"
```

x:y:z:(1) indicates evenly-weighted data. "help fit" for more details



Output

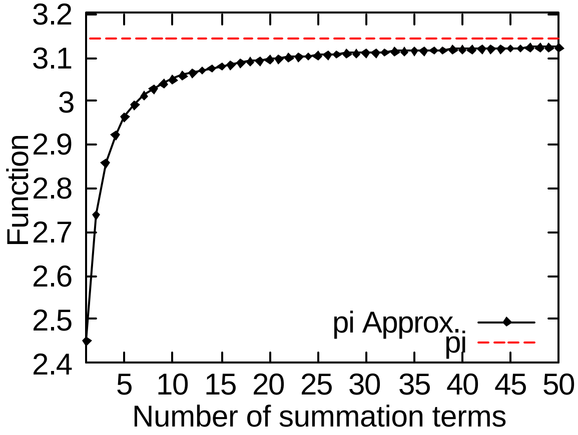
- Basic framework is

```
set terminal <TERMTYPE> <OPTIONS>    ← see "help set term"
set out "<FILENAME>"                 ← specifies output file
replot                               ← writes it to disk
set out                              ← closes the filehandle
set terminal windows|x11             ← restores the terminal
replot                               ← redraws the plot
```

- Most plots are automatically sized to fill a sheet of paper
  - Exceptions: encapsulated ps (more on this later), multiplot
  - So generally I preface this with
    - *set size 0.75,0.75* (or so, give or take)
  - Restore with
    - *set size 1,1*

Output (for windows)

- Windows wants a file that works with *Insert→Picture→From File*
  - For this talk, I used .png, the (free) alternative to .gif
    - *set term png small color*
  - For windows files that can be modified within PPT/Frame
    - *set term cgm color*
    - Then double-clicking converts it to a windows object
    - Functionally the same as right-click-copying from the display



Output (for real computers)

- Postscript terminal takes many options

```
                 [landscape]
                                           [color]  [solid]
set term post    [portrait]    [enhanced]                      [font] [size]
                                           [mono]   [dashed]
                    [eps]
```

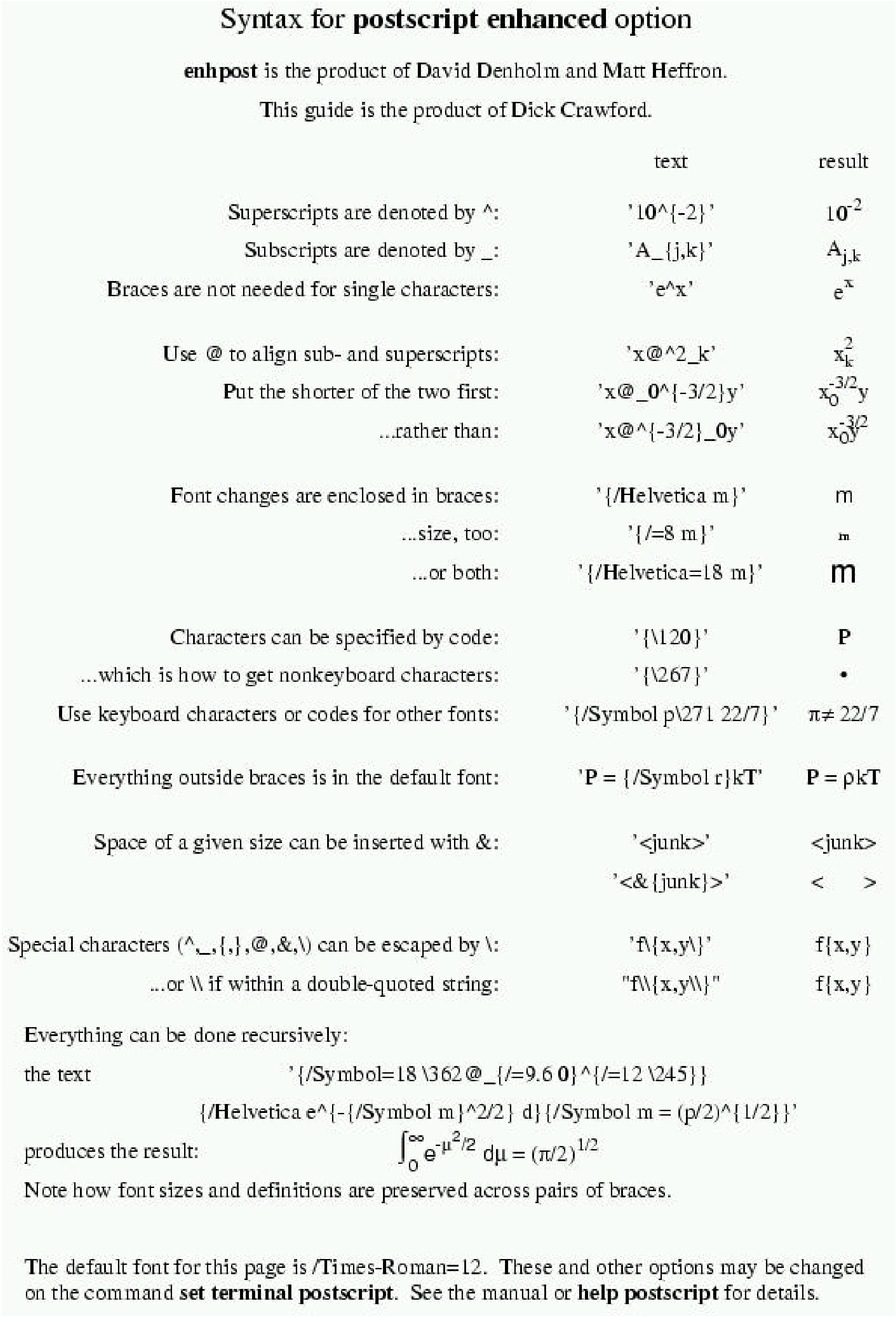
- *set term post eps enhanced color* is pretty standard fare
  - eps generates plots that are 5"x3.5"
    - set size 0.65,0.65 creates 1-LaTeX-column-sized plots
  - "Helvetica" 14 set by default; "Times-Roman" 14 decent, too
  - enhanced allows fancy texting in LaTeX-jargon (more later)

- Other possibilities include ("help set term latex")
  - fig: munging in xfig, then using transfig→ps or mifXfig→mif
  - latex, pslatex, pstex: direct incorporation into .tex files

Output with enhanced ps

From ps_guide.ps (comes with the gnuplot distro)

The handouts have the real pages



Syntax for **postscript enhanced** option

enhpost is the product of David Denholm and Matt Heffron.

This guide is the product of Dick Crawford.

	text	result
Superscripts are denoted by ^:	'10^{-2}'	10^{-2}
Subscripts are denoted by _:	'A_{j,k}'	$A_{j,k}$
Braces are not needed for single characters:	'e^x'	e^x
Use @ to align sub- and superscripts:	'x@^2_k'	x_k^2
Put the shorter of the two first:	'x@_0^{-3/2}y'	$x_0^{-3/2}y$
...rather than:	'x@^{-3/2}_0y'	$x^{-3/2}_{0}y$
Font changes are enclosed in braces:	'{/Helvetica m}'	m
...size, too:	'{/=8 m}'	m
...or both:	'{/Helvetica=18 m}'	m
Characters can be specified by code:	'{\120}'	P
...which is how to get nonkeyboard characters:	'{\267}'	•
Use keyboard characters or codes for other fonts:	'{/Symbol p\271 22/7}'	$\pi \neq 22/7$
Everything outside braces is in the default font:	'P = {/Symbol r}kT'	**P = ρkT**
Space of a given size can be inserted with &:	'<junk>'	<junk>
	'<&{junk}>'	< >
Special characters (^,_,{,},@,&,\) can be escaped by \:	'f\{x,y\}'	f{x,y}
...or \\ if within a double-quoted string:	"f\\{x,y\\}"	f{x,y}

Everything can be done recursively:

the text '{/Symbol=18 \362@_{/=9.6 0}^{/=12 \245}}

{/Helvetica e^{-{/Symbol m}^2/2} d}{/Symbol m = (p/2)^{1/2}}'

produces the result: $\int_0^\infty e^{-\mu^2/2}\, d\mu = (\pi/2)^{1/2}$

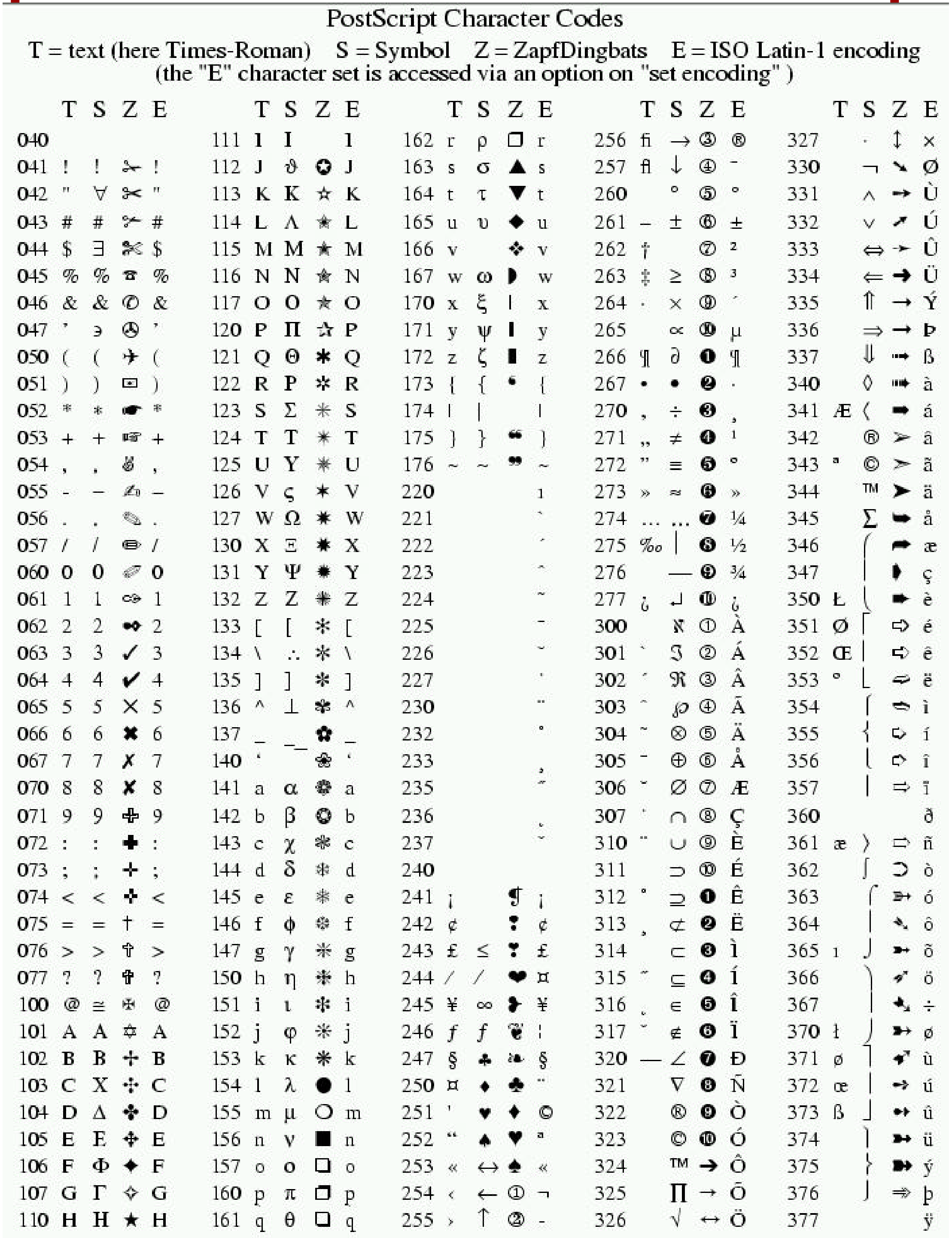
Note how font sizes and definitions are preserved across pairs of braces.

The default font for this page is /Times-Roman=12. These and other options may be changed on the command **set terminal postscript**. See the manual or **help postscript** for details.

Output with enhanced ps (con't)

From ps_guide.ps (comes with the gnuplot distro)

The handouts have the real pages



PostScript Character Codes

T = text (here Times-Roman) S = Symbol Z = ZapfDingbats E = ISO Latin-1 encoding
(the "E" character set is accessed via an option on "set encoding")

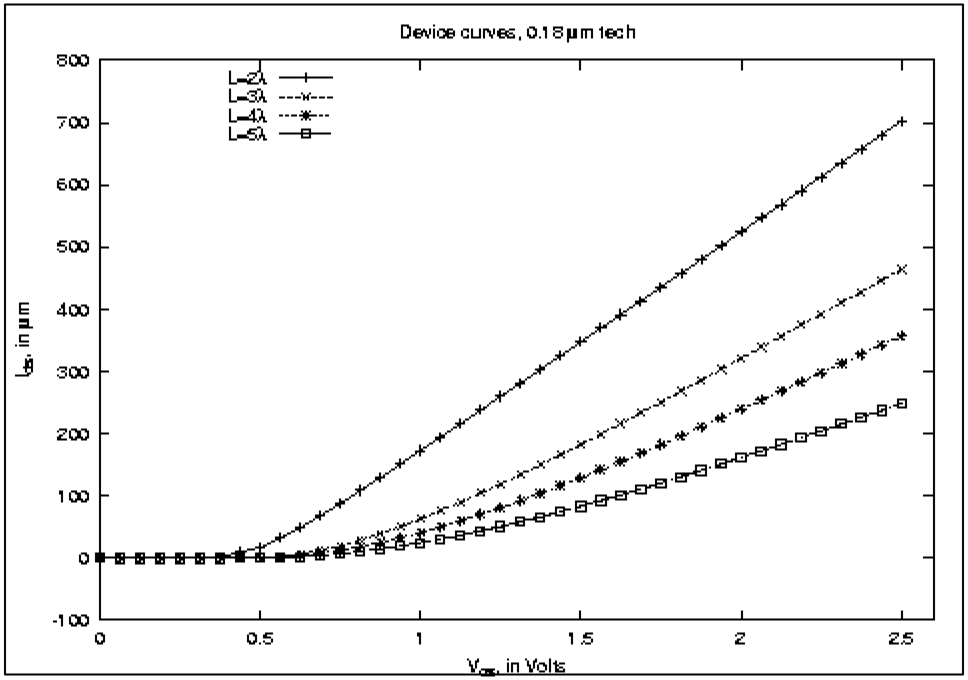
	T	S	Z	E		T	S	Z	E		T	S	Z	E		T	S	Z	E		T	S	Z	E
040					111	I	Ι	✩	I	162	r	ρ	❒	r	256	ﬁ	→	③	®	327		⋅	↕	×
041	!	!	✁	!	112	J	ϑ	✪	J	163	s	σ	▲	s	257	ﬂ	↓	④	¯	330		¬	➘	Ø
042	"	∀	✂	"	113	K	Κ	✫	K	164	t	τ	▼	t	260		°	⑤	°	331		∧	➙	Ù
043	#	#	✃	#	114	L	Λ	✬	L	165	u	υ	◆	u	261	–	±	⑥	±	332		∨	➚	Ú
044	$	∃	✄	$	115	M	Μ	✭	M	166	v		❖	v	262	†		⑦	²	333		⇔	➛	Û
045	%	%	☎	%	116	N	Ν	✮	N	167	w	ω	◗	w	263	‡	≥	⑧	³	334		⇐	➜	Ü
046	&	&	✆	&	117	O	Ο	✯	O	170	x	ξ	❘	x	264	·	×	⑨	´	335		⇑	➝	Ý
047	’	∋	✇	’	120	P	Π	✰	P	171	y	ψ	❙	y	265		∝	⑩	µ	336		⇒	➞	Þ
050	(	(	✈	(	121	Q	Θ	✱	Q	172	z	ζ	❚	z	266	¶	∂	❶	¶	337		⇓	➟	ß
051	)	)	✉	)	122	R	Ρ	✲	R	173	{	{	❛	{	267	•	•	❷	·	340		◊	➠	à
052	*	∗	☛	*	123	S	Σ	✳	S	174	|	|		|	270	‚	÷	❸	¸	341	Æ	〈	➡	á
053	+	+	☞	+	124	T	Τ	✴	T	175	}	}	❝	}	271	„	≠	❹	¹	342		®	➢	â
054	,	,	✌	,	125	U	Υ	✵	U	176	~	∼	❞	~	272	”	≡	❺	º	343	ª	©	➣	ã
055	-	−	✍	−	126	V	ς	✶	V	220				ı	273	»	≈	❻	»	344		™	➤	ä
056	.	.	✎	.	127	W	Ω	✷	W	221				`	274	…	…	❼	¼	345		∑	➥	å
057	/	/	✏	/	130	X	Ξ	✸	X	222				´	275	‰	|	❽	½	346		⎛	➦	æ
060	0	0	✐	0	131	Y	Ψ	✹	Y	223				ˆ	276		—	❾	¾	347		⎜	➧	ç
061	1	1	✑	1	132	Z	Ζ	✺	Z	224				˜	277	¿	↵	❿	¿	350	Ł	⎝	➨	è
062	2	2	✒	2	133	[	[	✻	[	225				¯	300		ℵ	➀	À	351	Ø	⎡	➩	é
063	3	3	✓	3	134	\	∴	✼	\	226				˘	301	`	ℑ	➁	Á	352	Œ	⎢	➪	ê
064	4	4	✔	4	135	]	]	✽	]	227				˙	302	´	ℜ	➂	Â	353	º	⎣	➫	ë
065	5	5	✕	5	136	^	⊥	✾	^	230				¨	303	ˆ	℘	➃	Ã	354		⎧	➬	ì
066	6	6	✖	6	137	_	_	✿	_	232				˚	304	˜	⊗	➄	Ä	355		⎨	➭	í
067	7	7	✗	7	140	‘	‾	❀	‘	233				¸	305	¯	⊕	➅	Å	356		⎩	➮	î
070	8	8	✘	8	141	a	α	❁	a	235				˝	306	˘	∅	➆	Æ	357		⎪	➯	ï
071	9	9	✙	9	142	b	β	❂	b	236				˛	307	˙	∩	➇	Ç	360				ð
072	:	:	✚	:	143	c	χ	❃	c	237				ˇ	310	¨	∪	➈	È	361	æ	〉	➱	ñ
073	;	;	✛	;	144	d	δ	❄	d	240					311		⊃	➉	É	362		∫	➲	ò
074	<	<	✜	<	145	e	ε	❅	e	241	¡		❡	¡	312	˚	⊇	➊	Ê	363		⌠	➳	ó
075	=	=	✝	=	146	f	φ	❆	f	242	¢		❢	¢	313	¸	⊄	➋	Ë	364		⎮	➴	ô
076	>	>	✞	>	147	g	γ	❇	g	243	£	≤	❣	£	314		⊂	➌	Ì	365	ı	⌡	➵	õ
077	?	?	✟	?	150	h	η	❈	h	244	⁄	⁄	❤	¤	315	˝	⊆	➍	Í	366		⎞	➶	ö
100	@	≅	✠	@	151	i	ι	❉	i	245	¥	∞	❥	¥	316	˛	∈	➎	Î	367		⎟	➷	÷
101	A	Α	✡	A	152	j	ϕ	❊	j	246	ƒ	ƒ	❦	¦	317	ˇ	∉	➏	Ï	370	ł	⎠	➸	ø
102	B	Β	✢	B	153	k	κ	❋	k	247	§	♣	❧	§	320	—	∠	➐	Ð	371	ø	⎤	➹	ù
103	C	Χ	✣	C	154	l	λ	●	l	250	¤	♦	♣	¨	321		∇	➑	Ñ	372	œ	⎥	➺	ú
104	D	Δ	✤	D	155	m	μ	❍	m	251	'	♥	♦	©	322		®	➒	Ò	373	ß	⎦	➻	û
105	E	Ε	✥	E	156	n	ν	■	n	252	“	♠	♥	ª	323		©	➓	Ó	374		⎫	➼	ü
106	F	Φ	✦	F	157	o	ο	❏	o	253	«	↔	♠	«	324		™	➔	Ô	375		⎬	➽	ý
107	G	Γ	✧	G	160	p	π	❐	p	254	‹	←	①	¬	325		∏	→	Õ	376		⎭	➾	þ
110	H	Η	★	H	161	q	θ	❑	q	255	›	↑	②	-	326		√	↔	Ö	377				ÿ

Output with enhanced ps (con't)

- Example of a plot, with labels redone to utilize symbols
  - – .eps with windows preview is large (50KB)

Changes:

```
set xlabel "V_{gs}, in Volts"
set ylabel "I_{ds}, in {/Symbol m}m"
set title "Device curves, 0.18 {/Symbol m}m tech"
plot "plot5.dat3" u ($1*2.5/2e-9):($2*-1e6) '%lf,%lf,%lf,%lf,%lf' t "L=2{/Symbol l}"
replot "plot5.dat3" u ($1*2.5/2e-9):($3*-1e6) '%lf,%lf,%lf,%lf,%lf' t "L=3{/Symbol l}"
replot "plot5.dat3" u ($1*2.5/2e-9):($4*-1e6) '%lf,%lf,%lf,%lf,%lf' t "L=4{/Symbol l}"
replot "plot5.dat3" u ($1*2.5/2e-9):($5*-1e6) '%lf,%lf,%lf,%lf,%lf' t "L=5{/Symbol l}"
set term post eps enhan color
set out "plot5.eps"; replot
set out; set term windows; replot
```



This is a crappy windows preview of an EPS; the printout looks much better

Interfacing with files

- Creating and loading command files
  - They are plain-text, so can create/edit with vi[m]/[x]emacs
  - Within gnuplot, create/use them with *save "file"*/*load "file"*
  - From shell, can call gnuplot with *gnuplot file*

- So far we've only plotted datafiles, but can also plot raw output
  - Within gnuplot, use, e.g.,
    - *plot "< simulator.pl" u 1:($2*1e9) t "ExecTime" w lp*
    - Although this usually optimizes the wrong resources…
  - This also allows constructs like
    - *plot "< awk '{print $1,sqrt($2*$3)}' foo.dat" u 1:2 t "Data" w lp*
    - Although *using $1:sqrt($2*$3)* does the same thing…
      - Plus, calling awk requires *popen()* support, which is missing under W2K…

Other odds and ends

- Using time on the xaxis (or yaxis, or zaxis):
  - *set xdata time; set timefmt "%Y/%m/%d.%H:%M:%S"*
    - … tells gnuplot what format your data x-col is in (*man date*)
- Too much data in your files to plot?
  - *plot "datafile" every 2*
    - … plots every other point. See "help every" for more details
- Want to plot a vertical line? (which isn't a function…)
  - *set arrow n from x1,y1 to x2,y2 nohead*
- gnuplot assumes integers unless you say so
  - 1/3 evaluates to 0; 1. /3. or 1.0/3.0 evaluates to 0.33…
    - … this burns me every other week

Conclusion

- I hope you learned something new about gnuplot

- Lots of sources for help
  - Introduction and FAQs on the web (do a search)
  - comp.graphics.apps.gnuplot and deja/google archives
  - Ask me (but if it's not covered here, I probably don't know…)

- By the way, www.cygwin.com has the tcsh environment for w32
  - It starts to make w32 a usable working environment