Bijlage 1
How Perl can help TEX

Wybo Dekker
wybo@servaly.hobby.nl

abstract
Perl may be an easy interface to TEX when it comes to repetitive tasks, like writing letters, creating reports from databases, and many more. This article shows how Perl can be used to generate many similar pictures *via* the *mfpic* style

keywords
perl,mfpic,mkpic

1 Introduction

I recently had to produce about 40 pictures for insertion into a book on elementary mathematics. I decided that the *mfpic* would suite most of my needs. But writing *mfpic* commands is not easy. Figure 1, for example, can be constructed using the following *mfpic* commands:

```
1   \mftitle{ce}
    \setlength{\mfpicunit}{1mm}
    \begin{mfpic}[16][5.45]{0}{4}{-6}{5}
    \axes
5   \hatchwd{2}
    \tlabel[bc](0,5.54){$y$}
    \tlabel[cl](4.21,0){$x$}
    \tlabel[tc](2,-0.18){\strut 2}
    \tlabel[bc](3,0.18){\strut 3}
10  \tlabel[cr](-0.07,-5){\strut -5}
    \tlabel[cr](-0.07,0){\strut 0}
    \tlabel[cr](-0.07,4){\strut 4}
    \rhatch\lclosed\connect
    \lines{(0,0),(0,4)}
15  \function{0,3,.05}{4-x*x}
    \lines{(3,-5),(3,0)}
    \endconnect
    \function{0,3.2,.05}{4-x*x}
    \dotted\arrow\lines{(3,-5),(0,-5)}
20  \dotted\arrow\lines{(3,-5),(3,0)}
    \tlabel[bc](3,4){\parbox[b]{60mm}{%
      \center $f(x)=4-x^2$}}
    \arrow\lines{(3,3.46),(1.7,1.1)}
    \tlabel[bc](2,5){\parbox[b]{60mm}{%
25    \center Area $O_1$}}
    \arrow\lines{(2,4.46),(1,2)}
    \tlabel[bc](4,2){\parbox[b]{60mm}{%
      \center Area $O_2$}}
    \arrow\lines{(4,1.46),(2.8,-2)}
30  \end{mfpic}
```

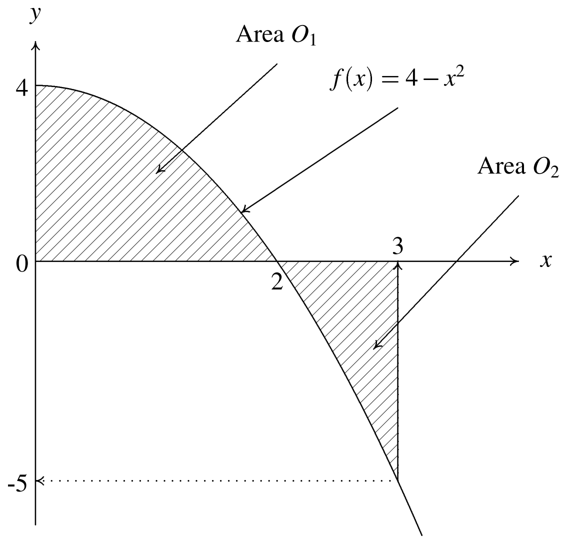

Figure 1. example a

As you can see, this implies a lot of typing and one has to type many nested [], {}, and () pairs. Also, several floating point numbers, such as those in lines 6–12, depend on the scaling factors defined in line 3. They have to be calculated manually, and changing the scale will imply recalculation of those values. The scale itself is set in line 3: I wanted the picture to be 64 mm wide, so I had to calculate $64/(4-0) = 16$ for the scaling factor in the x-direction. It would be much easier if one could type something like:

```
begin ce 64 64 0 -6 4 5 $x$ $y$
xmark 2
Xmark 3
ymark -5 0 4
bhat
lines 0 0 0 4
func 0 3 .05 4-x*x
lines 3 -5 3 0
ehat
func 0 3.2 .05 4-x*x
xydrop 3 -5
arrow 3 4 1.7 1.1 $f(x)=4-x^2$
arrow 2 5 1 2 Area $O_1$
arrow 4 2 2.8 -2 Area $O_2$
end
```

Here we see no brackets, braces or parentheses anymore, width and height are set straightforwardly to 64 mm and the labels along the axes are redefined as xmarks and ymarks, for which nothing has to be given but the x- and y-values, respectively. The corresponding y- and x-values are supposed to be calculated automatically.

Another construction that frequently occurs in my pictures is a label with an arrow starting from the center of its baseline, such as the one in lines 21–23 in the long listing. This is replaced in the short listing with line 12, where the starting position of the arrow is supposed to be calculated automatically. As a result, if I want to move the label, the arrow is moved with it automatically.

All this is possible by using a *Perl* interface that converts the short command file into an *mfpic* source file.

2 The Perl interface

I wrote the *Perl* script *mkpic* (section 5) on-the-fly: I first wrote lines 1–19 and 131–146, which just open, write and close files, define some handy variables and LaTeX commands, and print (line 130) anything in the input file literally to the *mfpic* output file. At that point, therefore, commands on the input file had to be valid *mfpic* commands. The initial lines also comprised a system call (line 134) running LaTeX, mf and xdvi, so that running the script would display the result. Instead of using a separate input file for my newly created commands, I put them in the `__DATA__` section of my script and read them from there. So I had to edit only one file for the creation of both my pictures and new commands.

Then, thinking about how I wanted my pictures to look, I inserted commands in lines 20–128 whenever I felt the need to define one. The first was the *begin* command, of course, which has also the most complex definition, as it defines many scale-dependent variables and TeX commands that might be useful for any command defined later.

Since I defined only what I needed, this *Perl* script does not have commands for every available *mfpic* command. But it is now easy to add more commands.

2.1 How to use mkpic

First of all, read the manpage of the *Perl*-script, generated from the script using pod2latex, which is shown in section 4.

The easiest way to use the script is to append your own commands to the `__DATA__` section of the script, and run it. This will produce a file mkpic.sty, which provides LaTeX-commands named `\Fig<name>`, where `<name>` stands for every name you use in the *begin* command. Finally, you can use those `\Fig<name>` commands in a LaTeX document.

3 Some more examples

Here are a few more examples illustrating some features of the *mkpic* script:

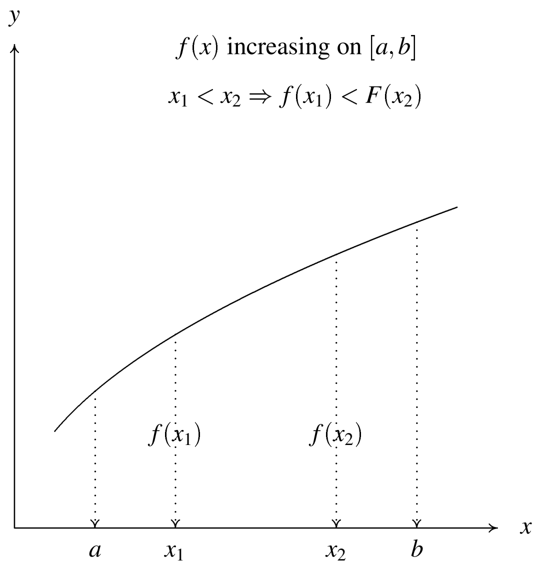

Figure 2. example b

The following commands will produce figure 2:

```
begin b 64 64 0 3 12 8 $x$ $y$
xmark $a$ 2 $x_1$ 4 $x_2$ 8 $b$ 10
ydrop 2 4.414
ydrop 4 5
ydrop 8 5.828
ydrop 10 6.162
label cc 4 4 $f(x_1)$
label cc 8 4 $f(x_2)$
label cc 7 8 $f(x)$ increases on $[a,b]$
label cc 7 7.5 $x_1<x_2\Rightarrow f(x_1)<F(x_2)$
func 1 11 .1 x**(.5)+3
end
```

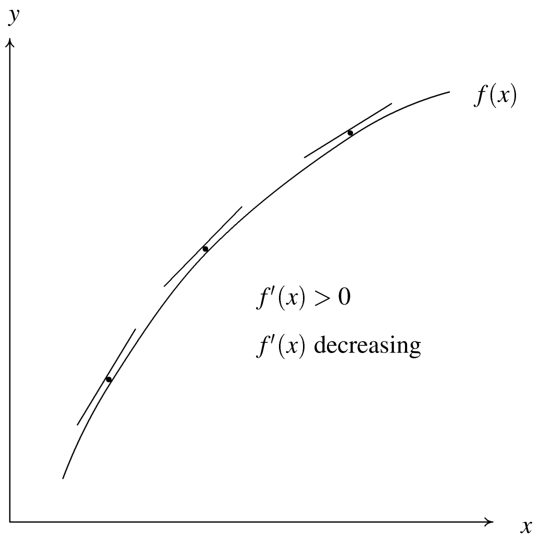

Figure 3. example c

These commands illustrate how valid *mfpic* commands can be interspersed between *mkpic* commands (see figure 3:

```
begin c 64 64 0 0 10 10 $x$ $y$
curve 1 1 2 3 4 5.7 7 8.1 9 9
\shift{(-.05,.05)}
point 2 3 4 5.7 7 8.1
\shift{(-.05,.05)}
func 1.4 2.6 .1 1.65*x-.3
func 3.2 4.8 .1 1.025*x+1.6
func 6.1 7.9 .1 .62*x+3.76
label cl 9.5 9 $f(x)$
label tl 5 5 $f^\prime(x)>0$
label tl 5 4 $f^\prime(x)$ decreasing
end
```

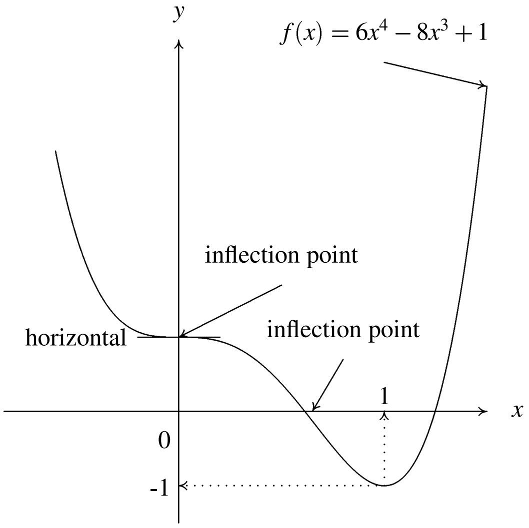

Figure 4. example d

Figure 4 is produced by:

```
begin d 64 64 -.85 -1.5 1.5 5 $x$ $y$
func -.6 1.5 .05 6*(x**4)-8*(x**3)+1
lines -.2 1 .2 1
label cr -.25 1 horizontal
arrow .5 2 0 1 inflection point
arrow .8 1 .65 0 inflection point
arrow 1 5 1.5 4.375 $f(x)=6x^4-8x^3+1$
Xmark 1
ymark \raisebox{-3.5mm}{0} 0 -1
xydrop 1 -1
end
```

And here is an elaborate quasi 3D picture. It shows how comments can be inserted. Standard axes are suppressed because they need special treatment (see figure 5):

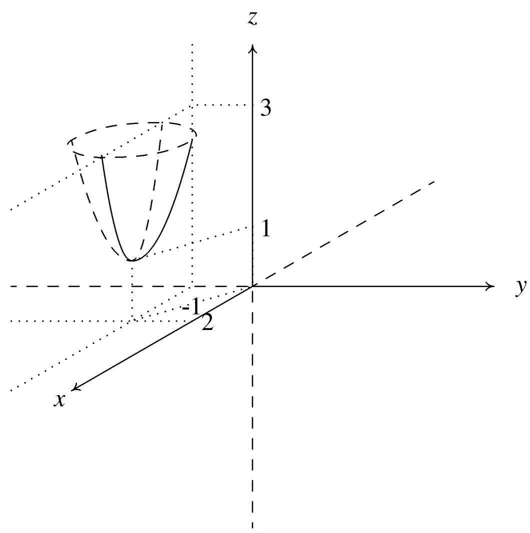

Figure 5. example e

```
begin e 64 64 -4 -4 4 4 - -
\dashed
lines -4 0 0 0 0 0 0 -4 # neg z and neg y
\dashed
lines 0 0 3 1.73        # neg x
\arrow
lines 0 0 0 4           # pos y
\arrow[5]
lines 0 0 4 0           # pos z
\arrow[5]
lines 0 0 -3 -1.73      # pos x
\dotted
lines -1 4 -1 0 -4 -1.73  # intersections y=-1 plane
\dotted
lines -1 -.577 -4 -.577 # intersection x=2 plane with xy-plane
% extra helplines
\dotted
lines -2 .423 -2 -.577 0 0 0 1 -2 .423
Ymark 3
% end of extras
\dotted
lines 0 3 -1 3 -4 1.27
\dashed\sclosed
curve -3 2.42 -1.5 2.711 -1 2.42 -2.5 2.134
label bc 0 \yhi $z$
label cl \xhi 0 $y$
label tr -3.1 -1.8 $x$
label cl -.85 -.577 2
label tc 0 5.5 $f(x,y)=x^2-4x+2y^2+4y+7$
xmark -1
Ymark 1
\shift{(-2,.42)}
\dashed
func 0 .5 .1 9*x*x
func -.5 0 .1 7*x*x
\dashed
func -1 0 .1 2*x*x
func 0 1 .1 2*x*x
end
```

4 The mkpic manpage

NAME

mkpic — interface for making pictures with mfpic

SYNOPSIS

```
mkpic [filename]
```

commands:

```
begin   name x1 y1 xmin ymin xmax ymax xlabel ylabel
xmark   [label1] x1 [label2] x2 ...
Xmark   [label1] x1 [label2] x2 ...
ymark   [label1] y1 [label2] y2 ...
Ymark   [label1] y1 [label2] y2 ...
xdrop   x y
ydrop   x y
xydrop  x y
arrow   x1 y1 x2 y2 label
label   YX x y label
point   x1 y1 x2 y2 ...
lines   x1 y1 x2 y2 ...
curve   x1 y1 x2 y2 ...
rect    x1 y1 x2 y2
crect   x1 y1 x2 y2
func    xmin xmax step expression-in-x
#       comment
hatch
bhat
ehat
end
stop
```

DESCRIPTION

mkpic provides an easy interface for generating commands for making small pictures with mfpic. To this end either an input file or a `__DATA__` section has to be created consisting of commands with space separated parameters. **filename** is the name of the input file; if no filename is given input is taken from the `__DATA__` section.

mkpic produces two files: **filename**`.sty`, which will contain LaTeX commands for every picture, and **filename**`.tex`, a LaTeX source file containing nothing but the pictures produced. This LaTeX source file will be compiled and displayed at the end of the script. If no **filename** is given, `mkpic.sty` will be used for the style file, and `mkpic.sty` for the LaTeX source file.

Currently the following commands are implemented:

begin end Every picture begins with the **begin** command and ends with the **end** command. The **begin** command defines a name for the picture and defines a latex `\newcommand` with that name, prefixed with 'Fig'. The resulting `\newcommand` is written to a .sty file. Thus the command

```
C<begin aa ...>
```

starts writing `\newcommand{\Figaa}{...` to the .sty file, and the picture can be reproduced in a LaTeX document by importing the .sty file and using the `\Figaa` command.
xl and yl are the lengths of the x- and y-axes. xlabel and ylabel are the label that are placed at the ends of those axes. Use a space to suppress labeling, or "−" to suppress drawing the axes at all.

xmark ymark Xmark Ymark These commands place one or more labels along the x- or y-axes, either below (**xmark** and **ymark**) of above (**Xmark** and **Ymark**) the axis.
For the `[`**xXyY**`]`**mark** commands a parameter containing any character other than `[-.0−9]` is interpreted as the label to be placed and its position is expected in the next parameter. If a parameter is just a number, it is placed at that x-position.

xdrop ydrop xydrop These commands draw dotted arrows perpendicularly to the x-axis, the y-axis and both axes, respectively, ending on the axes with the arrow head.

arrow draws an arrow from *(x1,y1)* to *(x2,y2)* labeled on its tail with *label*

label draws a label at *(x,y)*. *YX* tells how it will be adjusted: for Y=t,b,c *(x,y)* will be, in the y-direction, on top, bottom or center of the label respectively, for X=l,r,c it will be, in the x-direction, left, right or center adjusted on *(x,y)*. Thus

```
C<label tl 0 0 Hello>
```

will draw the string Hello with its lower left corner at (0,0)

point draws points (dots) at (x1,y1), (x2,y2) etcetera.

lines draws line segsments from (x1,y1) to (x2,y2), *(x3,y3)* etcetera.

curve draws a bezier curve from (x1,y1) to (x2,y2), (x3,y3) etcetera.

rect draws a rectangle with diagonal points at (x1,y1) and (x2,y2).

crect clears a rectangle with diagonal points at (x1,y1) and (x2,y2).

func draws the function given by *expression-in-x* between *xmin* and *xmax*, stepping with *step* units in the x-direction.

# denotes a comment

hatch hatch the closed curve that follows.

bhat starts a path that will eventually be closed, and then hatched.

ehat ends a path started with **bhat**, closes it and then hatches it.

anything else will be inserted as is in the style file, and therefore should be a valid *mfpic* statement. You use this when you need such a staement only once, or a few times and therefore see no need to define a proper command for it.

stop stops further reading of the input. Useful if you have many pictures, but want to see only the first few for testing purposes.

5 The Perl script

This is the *Perl*-script without the pod-text. I removed it as the manpage is shown in a separate section:

```
#!/usr/bin/perl -w
use vars qw($com);

$tex=shift;
if ($tex) { open(IN,$tex); $input=*IN }
      else { $input=*DATA; $tex='mkpic'; }

open_stylefile();
for (glob('pictures.*')) {unlink $_}

open(TEX,">$tex.tex");
print TEX '\documentclass[a4paper]{report}
\usepackage{'.$tex.'}
\begin{document}\noindent
';

%pos=('x'=>'tc',        # positions for [xyXY]marks
      'y'=>'cr',
      'X'=>'bc',
      'Y'=>'cl'
     );

while (<$input>) {
  chomp;
  s/\s*#.*//;            # remove comment
  next unless $_;        # skip empty lines
  /^begin/ and do {
    ($com,$name,$xl,$yl,$xmin,$ymin,
     $xmax,$ymax,$xlabel,$ylabel)=split;
    $xlabel="" if $xlabel eq '-';
    $ylabel="" if $ylabel eq '-';
    $xscale=int(100*$xl/($xmax-$xmin)+.5);
    $yscale=int(100*$yl/($ymax-$ymin)+.5);
    $dx=sprintf("%.2f",100/$xscale);
    $dy=sprintf("%.2f",100/$yscale);
    $yx=$ymin>0 ? $ymin : 0; # y-position of the x-axis
    $xy=$xmin>0 ? $xmin : 0; # x-position of the y-axis
    $xlo=$xy-$dx;    # x-pos of right side of y-markers
    $ylo=$yx-$dy;    # y-pos of top side of x-markers
    $xhi=$xmax+3*$dx; # x-pos of left side of x-label
    $yhi=$ymax+3*$dy; # y-pos of bottom side of y-label
    print
      "%\n%====$name====\n".
      "\\newcommand{\\Fig$name}{\\mftitle{$name}\n".
      "\\def\\xlo{$xlo}\\def\\xhi{$xhi}\n".
      "\\def\\ylo{$ylo}\\def\\yhi{$yhi}\n".
      "\\def\\xy{$xy}\\def\\yx{$yx}\n".
      "\\vspace*{5ex}\n".
      "\\begin{mfpic}[$xscale][$yscale]".
      "{$xmin}{$xmax}{$ymin}{$ymax}\n".
      "\\hatchwd{2}\n".
    # "\\tlabel[tr]($xmin,$ylo){\\fbox{$name}}\n".
      "\\tlabel[bc]($xy,$yhi){$ylabel}\n".
      "\\tlabel[cl]($xhi,$yx){$xlabel}\n";
    print TEX "\\mbox{\\Fig$name}\\\\[20mm]\n";
},next;
/^arrow/ and do {
  ($com,$x1,$y1,$x2,$y2,$label)=split(/\s+/,$_,6);
  print "\\tlabel[bc]($x1,$y1){\\parbox[b]{60mm}".
        "{\\center $label}}\n".
        "\\arrow\\lines{($x1,".($y1-$dy*3)."),($x2,$y2)}\n";
},next;
/^[xyXY]mark/ and do {
  ($_,@z)=split;
  s/mark//;   # 'x', 'y', 'X' or 'Y'
  for ($i=0;$i<@z;$i++) {
    ($label=$z[$i]) =~ /^[-.\d]+$/ or $i++;
    $x=/x/i ? $z[$i] : /y/ ? $xlo : -$xlo;
    #   ^xmark?        ymark?       Ymark!
    $y=/y/i ? $z[$i] : /x/ ? $ylo : -$ylo;
    #   ^ymark?        xmark?       Xmark!
    print "\\tlabel[$pos{$_}]($x,$y)".
          "{\\strut $label}\n";
  }
},next;
/^(point|lines|curve|rect|crect)/ and do {
                  # for example: crect 5 20 20 5
  s/^/\\/;        # \crect 5 20 20 5
  s/\s+/{(/;      # \crect{(5 20 20 5
  while(/\s+/) {
    s/\s+([-.\d]+) ?/,$1),(/;
  }               # \crect{(5,20),(20,5)
  s/..$/}/;       # \crect{(5,20),(20,5)}
  s/crect/gclear\\rect/;
                  # \gclear\rect{(5,20),(20,5)}
  print "$_\n";
},next;
/^func/ and do {
  ($com,$x,$y,$d,$f)=split;
  print "\\function{$x,$y,$d}{$f}\n";
},next;
/^hatch/ and do {
  print '\rhatch\draw\lclosed';
},next;
/^bhat/ and do {
  print '\rhatch\lclosed\connect',"\n";
},next;
/^ehat/ and do {
  print '\endconnect',"\n";
},next;
/^xdrop/ and do {
  ($com,$x,$y)=split;
  print "\\dotted\\arrow".
        "\\lines{($x,$y),($xy,$y)}\n";
},next;
/^ydrop/ and do {
  ($com,$x,$y)=split;
  print "\\dotted\\arrow".
        "\\lines{($x,$y),($x,$yx)}\n";
},next;
/^xydrop/ and do {
  ($com,$x,$y)=split;
  print "\\dotted\\arrow".
        "\\lines{($x,$y),($xy,$y)}\n";
  print "\\dotted\\arrow".
```

```
          "\\lines{($x,$y),($x,$yx)}\n";
  },next;
  /^label/ and do {
    ($com,$m,$x,$y,$label)=split(/\s+/,$_,5);
    $m=~/^[bct][lcr]$/ or die "illegal label in $_\n";
    print "\\tlabel[$m]($x,$y){$label}\n";
  },next;
  /^end/ and do {
    # axes drawn last for easier rect clears:
    print "\\arrow[\\axisheadlen]\\lines".
          "{($xmin,$yx),($xmax,$yx)}\n" if $xlabel;
    print "\\arrow[\\axisheadlen]\\lines".
          "{($xy,$ymin),($xy,$ymax)}\n" if $ylabel;
    print "\\end{mfpic}}\n";
  },next;
  /^stop/ and do { last }; # stop reading the input
  print "$_\n" if $_; # anything else printed literally
}
print TEX "\\end{document}\n";
close(TEX);
close(STY);
system("latex $tex && \
        mf '\\mode=localmode;' input pictures && \
        latex $tex && \
        xdvi $tex");
sub open_stylefile {
open(STY,">$tex.sty"); select STY;
print '\RequirePackage{ifthen}
\input mfpic
\AtBeginDocument{\opengraphsfile{pictures}}
\AtEndDocument{\closegraphsfile}
\setlength{\mfpicunit}{.01mm}
';
}
__DATA__
begin a 64 64 0 -6 4 5 $x$ $y$
xmark 2
Xmark 3
ymark -5 0 4
bhat
lines 0 0 0 4
func 0 3 .05 4-x*x
lines 3 -5 3 0
ehat
func 0 3.2 .05 4-x*x
xydrop 3 -5
arrow 3 4 1.7 1.1 $f(x)=4-x^2$
arrow 2 5 1 2 Area $O_1$
arrow 4 2 2.8 -2 Area $O_2$
end

begin b 64 64 0 3 12 8 $x$ $y$
xmark $a$ 2 $x_1$ 4 $x_2$ 8 $b$ 10
ydrop 2 4.414
ydrop 4 5
ydrop 8 5.828
ydrop 10 6.162
label cc 4 4 $f(x_1)$
label cc 8 4 $f(x_2)$
label cc 7 8 $f(x)$ increasing on $[a,b]$
label cc 7 7.5 $x_1<x_2\Rightarrow f(x_1)<F(x_2)$
func 1 11 .1 x**(.5)+3
end
begin c 64 64 0 0 10 10 $x$ $y$
curve 1 1 2 3 4 5.7 7 8.1 9 9
\shift{(-.05,.05)}
point 2 3 4 5.7 7 8.1
\shift{(-.05,.05)}
func 1.4 2.6 .1 1.65*x-.3
func 3.2 4.8 .1 1.025*x+1.6
func 6.1 7.9 .1 .62*x+3.76
label cl 9.5 9 $f(x)$
label tl 5 5 $f^\prime(x)>0$
label tl 5 4 $f^\prime(x)$ decreasing
end

begin d 64 64 -.85 -1.5 1.5 5 $x$ $y$
func -.6 1.5 .05 6*(x**4)-8*(x**3)+1
lines -.2 1 .2 1
label cr -.25 1 horizontal
arrow .5 2 0 1 inflection point
arrow .8 1 .65 0 inflection point
arrow 1 5 1.5 4.375 $f(x)=6x^4-8x^3+1$
Xmark 1
ymark \raisebox{-3.5mm}{0} 0 -1
xydrop 1 -1
end

begin e 64 64 -4 -4 4 4 - -
\dashed
lines -4 0 0 0 0 0 0 -4  # neg z and neg y
\dashed
lines 0 0 3 1.73         # neg x
\arrow
lines 0 0 0 4            # pos z
\arrow
lines 0 0 4 0            # pos y
\arrow
lines 0 0 -3 -1.73       # pos x
\dotted
lines -1 4 -1 0 -4 -1.73 # intersections y=-1 plane
\dotted
lines -1 -.577 -4 -.577  # intersection x=2 plane with xy-plane
% extra helplines
\dotted
lines -2 .423 -2 -.577 0 0 0 1 -2 .423
Ymark 3
% end of extras
\dotted
lines 0 3 -1 3 -4 1.27
\dashed\sclosed
curve -3 2.42 -1.5 2.711 -1 2.42 -2.5 2.134
label bc 0 \yhi $z$
label cl \xhi 0 $y$
label tr -3.1 -1.8 $x$
label cl -.85 -.577 2
label tc 0 5.5 $f(x,y)=x^2-4x+2y^2+4y+7$
xmark -1
Ymark 1
\shift{(-2,.42)}
\dashed
func 0 .5 .1 9*x*x
func -.5 0 .1 7*x*x
\dashed
func -1 0 .1 2*x*x
func 0 1 .1 2*x*x
end
```