The Calcplot Package for Maple V[†]

Timothy A. Murdoch
Department of Mathematics
Washington and Lee University
Lexington, Virginia 24450
E-mail: `murdoch@wlu.edu`

August 8, 1994

Introduction

The contents of this document describe, in brief, the procedures available in the `calcplot` package, version 1.0. The package, including the source file and .dvi file for this document, and a *Maple* worksheet containing the examples found in this manual, is available on 3.5" floppy disk from the author by mail for two dollars US (to cover the cost of the disk and postage) at the above address. Please specify platform type in your correspondence. The package is also available via anonymous ftp at `ftp.liberty.wlu.edu` in the directory `/pub/math/maple/calcplot`. Any questions or bug reports will be welcomed, as will suggestions for improvement and enhancement of the package, and should be directed to the author at the e-mail address above. Anyone contacting the author about the package will be added to a mailing list for announcements of new versions/updates of the calcplot package.

The procedures in the `calcplot` package are an enhancement and extension of procedures found in the `mvcal` package of Cheung and Harer. The author expresses thanks to both Cheung and Harer for the inspiration afforded by the ideas in their package and for making their code available to him. Thanks also to Robert Lopez for encouragement, criticism, and classroom testing.

Maple Procedures

The names of the *Maple* procedures defined in the `calcplot` package, along with a brief description, are provided in this section. Examples of particular usage are given in the next section.

Each procedure in the `calcplot` package returns either a `plot` structure (2D plot) or a `plot3d` structure (3D plot). Thus the output of different procedures can be combined, provided the outputs have the same plot structure. See the help page for `display` and `display3d` for more information about combining plots.

Each procedure accepts plotting options; that is, the user may specify any plotting option that a plot of a similar type (either `plot` or `plot3d`) will accept. For example, a procedure that returns a `plot3d` structure can be given an option such as `style = WIREFRAME` to override the defaults set by `calcplot` . When the `calcplot` defaults are overridden, all other plotting options set by `calcplot` are overridden and set to the *Maple* defaults. The defaults set by `calcplot` are:

1) the options `axes=NORMAL`, `scaling=CONSTRAINED`, and, in some cases, the `grid` option, for procedures that return `plot` structures.
2) the options `axes=BOXED`, `scaling=CONSTRAINED`, `style=PATCH`, and, in some cases, the `grid` and `label` options, for procedures that return `plot3d` structures.

Note, however, that it is possible to add options to a plot (instead of resetting them all) by using the `display` or `display3d` procedures from the `plots` package. See the second entry in Examples.

[†] Supported in part by NSF grant DUE-9351272

• Surface of rotation plotters. Each of these procedures returns a `plot3d` structure. Moreover, the procedures accept `plot3d` plotting options (as the optional *opts* argument below), and different plots can be combined with the `display3d` procedure from the `plots` package. Note that the names are consistent with the descriptions for the x-axis as $y = 0$ and the y-axis as $x = 0$.

`rotxplot` Syntax: `rotxplot(f,x=a..b,y=c,`*opts*`)`

This procedure plots the surface of rotation generated by rotating a curve in $\mathbf{R}^2$ about a user-specified axis that is parallel to the x-axis. The curve, represented by `f`, may be given as an expression, a procedural function (a function defined by an arrow), or, for the case of a parametric curve, a two-element list of expressions. The second argument gives the domain of the independent variable x, and the last argument gives the axis of rotation.

`rotyplot` Syntax: `rotyplot(f,x=a..b,x=c,`*opts*`)`

This procedure plots the surface of rotation generated by rotating a curve in $\mathbf{R}^2$ about a user-specified axis that is parallel to the y-axis. The syntax is identical to that of `rotxplot` except that the third argument is of the form `x=c`.

• Procedures to plot 2D regions. Each of these procedures returns a `plot` structure. The procedures accept `plot` options as the (optional) *opts* argument. Separate plots may be combined with the `display` procedure from the `plots` package.

`dydxplot` Syntax: `dydxplot(y=f(x)..g(x),x=a..b,`*opts*`)`

This procedure plots the region in $\mathbf{R}^2$ between two graphs $y = f(x)$ and $y = g(x)$ for $a \le x \le b$. In addition, three vertical lines passing through the interior of the region are drawn. A companion procedure, `dxdyplot` plots a region in $\mathbf{R}^2$ between two graphs $x = h(y)$ and $x = k(y)$ for $c \le y \le d$. The syntax is entirely similar; note only that the roles of x and y are reversed. Moreover, the syntax of these procedures is the same as that required by *Maple* to perform a double integral over the region. These procedures were inspired by procedures of the same name in the `mvcal` package of Cheung and Harer.

`drdtplot` Syntax: `drdtplot(r=f(theta)..r=g(theta),theta=a..b,`*opts*`)`

This procedure plots the region in $\mathbf{R}^2$ between the two polar graphs $r = f(\theta)$ and $r = g(\theta)$ for $a \le \theta \le b$. In addition, four radial segments passing through the interior of the region are drawn. A companion procedure, `dtdrplot` plots a region in $\mathbf{R}^2$ between two graphs $\theta = h(r)$ and $\theta = k(r)$ for $c \le r \le d$. The syntax is entirely similar; note only that the roles of θ and r are reversed. Moreover, the syntax of these procedures is the same as that required by *Maple* to perform a double integral over the region. These procedures were inspired by procedures of the same name in the `mvcal` package of Cheung and Harer.

• Procedures to plot graphs of functions of two variables over non-rectangular regions. Each of these procedures returns a `plot3d` structure, accepts optional `plot3d` options, and can be combined with other `plot3d` structures with the `display3d` procedure from the `plots` package.

`yxgraphplot` Syntax: `yxgraphplot(q(x,y),y=f(x)..g(x),x=a..b,`*opts*`)`

This procedure plots the graph $z = q(x, y)$ over the region in the $\mathbf{R}^2$ between two graphs $y = f(x)$ and $y = g(x)$ for $a \leq x \leq b$. The function $q(x, y)$ is allowed to be either an expression or a procedural function (a function defined by an arrow) taking two arguments. A companion procedure, `xygraphplot` plots the graph $z = q(x, y)$ over the region in $\mathbf{R}^2$ between two graphs $x = h(y)$ and $x = k(y)$ for $c \leq y \leq d$. The syntax is entirely similar; note only that the roles of x and y are reversed. The syntax of both procedures reflects their similarity to `dydxplot` and `dxdyplot`. Note also that these plots may be rendered by *Maple*'s `plot3d` procedure. In the interest of students' ease of use, we include redundant capabilities in this package.

`rtgraphplot` Syntax: `rtgraphplot(Q(r,theta),r=f(theta)..g(theta),theta=a..b,`*opts*`)`

This procedure plots the graph $z = Q(r, \theta)$ over the region in the $\mathbf{R}^2$ between two graphs $r = f(\theta)$ and $r = g(\theta)$ for $a \leq \theta \leq b$. The function $Q(r, \theta)$ is allowed to be either an expression or a procedural function (a function defined by an arrow) taking two arguments. A companion procedure, `trgraphplot` plots the graph $z = Q(r, \theta)$ over the region in $\mathbf{R}^2$ between two graphs $\theta = h(r)$ and $\theta = k(r)$ for $c \leq r \leq d$. The syntax is entirely similar; note only that the roles of θ and r are reversed. The syntax of both procedures reflects their similarity to `drdtplot` and `dtdrplot`.

- Procedures to plot 3-dimensional regions of integration.

This collection of procedures includes eighteen different plotting procedures. There are six procedures for each coordinate type description: cartesian coordinates (the usual (x, y, z) coordinates), cylindrical coordinates, (r, θ, z), and spherical coordinates (ρ, θ, ϕ). In each coordinate system, the procedures are similar enough to one another that only one procedure will be described. Each of these plots returns a `plot3d` structure, accepts optional `plot3d` options as the *opts* argument. The author hopes that an understanding of the syntax for the procedures described in this section will allow the reader to determine the correct syntax for the other fifteen procedures not explicitly described here.

`dzdydxplot` Syntax: `dzdydxplot(z=f(x,y)..g(x,y),y=h(x)..k(x),x=a..b,`*opts*`)`

This procedure plots (the boundary of) the region in $\mathbf{R}^3$ that is, when described in cartesian coordinates, between the graphs $z = f(x, y)$ and $z = g(x, y)$ and lies over the region where $h(x) \leq y \leq k(x)$ and $a \leq x \leq b$. The syntax of these procedures is the same as that required by *Maple* to perform a triple integral over the region in the desired order. The five companion procedures are named `dzdxdyplot, dxdzdyplot, dydzdxplot, dxdydzplot, dydxdzplot`. The syntax for each of these procedures is entirely similar to that of `dzdydxplot`. These procedures, too, were inspired by procedures of the same name in the `mvcal` package of Cheung and Harer.

`dzdrdtplot` Syntax:
`dzdrdtplot(z=f(r,theta)..g(r,theta), r=h(theta)..k(theta), theta=a..b,`*opts*`)`

This procedure plots (the boundary of) the region in $\mathbf{R}^3$ that is, when described in cylindrical coordinates (r, θ, z), between the graphs $z = f(r, \theta)$ and $z = g(r, \theta)$ and lies over the region where $h(\theta) \leq r \leq k(\theta)$ and $a \leq \theta \leq b$. The syntax of these procedures is the same as that required by *Maple* to perform a triple integral over the region in the desired order. The five companion procedures are named `dzdtdrplot, drdzdtplot, drdzdtplot, dtdrdzplot, dtdzdrplot`. The syntax for each of these procedures is entirely similar to that of `dzdrdtplot`.

`dpdtdphiplot` Syntax:
`dpdtdphiplot(rho=f(theta,phi)..g(theta,phi), theta=h(phi)..k(phi), phi=a..b,`*opts*`)`

This procedure plots (the boundary of) the region in $\mathbf{R}^3$ that is, when described in spherical coordinates (ρ, θ, ϕ), determined by the inequalities $f(\theta, \phi) \leq \rho \leq g(\theta, \phi)$, $h(\phi) \leq \theta \leq k(\phi)$, and $a \leq \phi \leq b$. The syntax of these procedures is the same as that required by *Maple* to perform a triple integral over the region in the desired order. The five companion procedures are named `dpdphidtplot, dtdpdphiplot, dtdphidpplot, dphidtdpplot, dphidpdtplot`. The syntax for each of these procedures is entirely similar to that of `dpdtdphiplot`.

Finally, the `calcplot` package comes with on-line help. Get started with `?calcplot`.

Examples

Here we illustrate the use of the calcplot package with some specific examples. The reader is strongly encouraged to try all of theses examples to gain familiarity with the syntax, as well as to test the installation and operation of the package itself.

For surfaces of rotation about an axis parallel to the y-axis try the following:

```
>rotyplot(x^2,x=0..1,x=1);
```

```
>plots[display3d](rotyplot(x->x^2,x=0..1,x=2),title=`A spun parabola`);
```

Note that either an expression or a function may be used as the first argument. Note also that parametic curves can be used.

```
>rotyplot([cos(x),sin(x)],x=0..Pi,x=3);
```

Next, some examples of surfaces of rotation about an axis parallel to the x-axis.

```
>rotxplot(x^2,x=0..1,y=1);

>rotxplot(sin(x),x=0..2*Pi,y=0);

>rotxplot(x->sin(x),x=0..2*Pi,y=-1);

>rotxplot([cos(x),sin(x)],x=0..Pi,y=-2);
```

The examples in the next group show plots of domains for double integrals in cartesian and polar coordinates. These procedures return `plot` structures and so may be combined with the `display` command from the `plots` package.

```
>dydxplot(y=x^2..x+1,x=0..1);

>dxdyplot(x=sqrt(y)..y^2,y=1..2);

>drdtplot(r=1..1+cos(theta),theta=-Pi/2..Pi/2);

>dtdrplot(theta=sqrt(r)..Pi/2,r=0..1);
```

The next examples are plots of graphs of functions over non-rectangular planar domains. Note that the first argument for these procedures can be either an expression or a function. These procedures return `plot3d` stuctures, so letting the first argument be zero will render a `plot3d` image of the domain. Also, the output from these procedures may be combined using the `display3d` command from the `plot3d` package.

```
>yxgraphplot(x*y,y=0..sqrt(1-x^2),x=-1..1);

>xygraphplot(x*y,x=-sqrt(1-y^2)..sqrt(1-y^2), y=0..1);

>yxgraphplot((x,y)->sin(Pi*(x^2+y^2)),y=x^2..x^(1/4),x=0..1);

>rtgraphplot(r^2*sin(theta)*cos(theta),r=1..1+cos(theta),theta=-Pi/2..Pi/2);

>rtgraphplot((r,theta)->theta,r=sin(3*theta)..sin(theta),theta=Pi/4..Pi/3);
```

A more representative view of the previous plot can be obtained with the command:

```
>rtgraphplot((r,theta)->theta,r=sin(3*theta)..sin(theta),theta=Pi/4..Pi/3,view=0..1.1);
```

The examples below show plots of triple-integral domains in cartesian, cylindrical and spherical coordinates. Each procedure returns a `plot3d` structure and so separate plots may be combined using the `display3d` command from the `plots` package. Note also that the `display3d` procedure can be used to change some of the `plot3d` options set by the `calcplot` procedures; in particular, see the fourth example below.

```
>dzdydxplot(z=x^2+y^2..2,y=0..x,x=0..1,style=PATCH,orientation=[30,70],axes=NORMAL,
 scaling=CONSTRAINED,title=`MTW page 316, the right way`);

>drdtdzplot(r = 2..3, theta = Pi/4..Pi/2,z = 0..1);

>dzdrdtplot(z=0..sqrt(4-r^2),r=0..1+cos(theta),theta=0..2*Pi);

>p1:=dzdrdtplot(z=0..sqrt(1-r^2),r=0..1,theta=-Pi/2..Pi/2):
 p2:=dzdrdtplot(z=0..sqrt(1-r^2),r=0..1+cos(theta),theta=Pi/2..3*Pi/2):
 plots[display3d]({p1,p2});

>dpdtdphiplot(rho=0..1,theta=0..2*Pi,phi=0..5*Pi/12);

>P1:=dpdtdphiplot(rho=sec(phi)..2/(cos(phi)+sin(phi)),theta=0..2*Pi, phi=0..Pi/4):
 P2:=dpdtdphiplot(rho=0..cot(phi)*csc(phi),theta=0..2*Pi,phi=Pi/4..Pi/2):
 plots[display]({P1,P2},style=WIREFRAME);
```

In addition to the surface of rotation plotters, the triple-integral-region plotters can be used for demonstation purposes too. We give two examples below from second-semester calculus: solids whose base is the region in the xy-plane given by $x^2 \le y \le 8 - x^2,\ -2 \le x \le 2$ such that the cross-section of the solid perpendicular to the x-axis is a square, respectively, a semi-circle. (The alert reader will notice that two of the three plots given are for the same region.)

```
>dzdydxplot(z=0..8-2*x^2,y=x^2..8-x^2,x=-2..2,title=`Square cross-sections`);

>dzdydxplot(z=0..sqrt((4-x^2)^2-(y-4)^2),y=x^2..8-x^2,x=-2..2,
 title=`Semi-circular cross-sections (almost)` );

>dzdydxplot(z=0..(4-x^2-y^2)^(0.5),y=x^2..8-x^2, x=-2..2, title=`Full Semi-circles`);
```

History

1. Version 1.0 finished in August, 1994.