Exploring the Mathematics Behind Skateboarding: Analysis of the Directional Derivative

Note: You may notice differences between this Maple worksheet and the equivalent Mathematica notebook. These differences were introduced to preserve the content of these modules and were necessary because of major functional differences between Maple and Mathematica.

Introduction

OBJECTIVE: You will see how the directional derivative changes as you traverse a path through the domain of a function $z = f(x,y)$.

What is a directional derivative, and what does it look like? To find out, you get to put yourself in the position of a skateboarder and explore skating on a flat ramp or inside a bowl. The graphical representation of the dot product of the gradient of the surface function and a unit vector in the direction of motion will become clearer. You will be able to plot the directional derivative as a function of the parameter t.

Technology Guidelines

NOTE: If you have just finished a worksheet, **restart** *Maple* before executing a new worksheet.
TO OPEN SECTIONS,
Click on the **PLUS** sign at the left hand side of the screen *or* select **Expand All Sections** from the **View** drop down menu.

TO STOP AN EXECUTION
Click on **STOP** button from the toolbar.

ORDER OF EXECUTION
Execute commands in the order given. Do not skip any *Maple* Input lines within a given worksheet

Alternatively, you can execute the entire worksheet by selecting the **Execute Worksheet** command from the **Edit** drop down menu.

SAVING WORKSHEETS.

You can save anytime to any directory you choose, and it is wise to save often.
EXPERIENCING MAJOR PROBLEMS
Save if appropriate, and then shut down *Maple* and start it up again.

Part I: Skateboarding on a Flat Horizontal Surface

Imagine a skateboarder tracing out a figure-8 on a horizontal plane. The first set of commands that follow give the parametric equations of the skateboarder's path in the *xy*-plane. Start with the equations in polar form.

```
> restart:
```

```
> r:=t->16*sin(t)^2;
  theta:=t;
```

$$r := t \to 16 \sin(t)^2$$

$$\theta := t$$

You need to read in the following **with(plots)** package to do polar plots. Be sure to read it in BEFORE you try to execute the plots, and remember, to read it in only once.

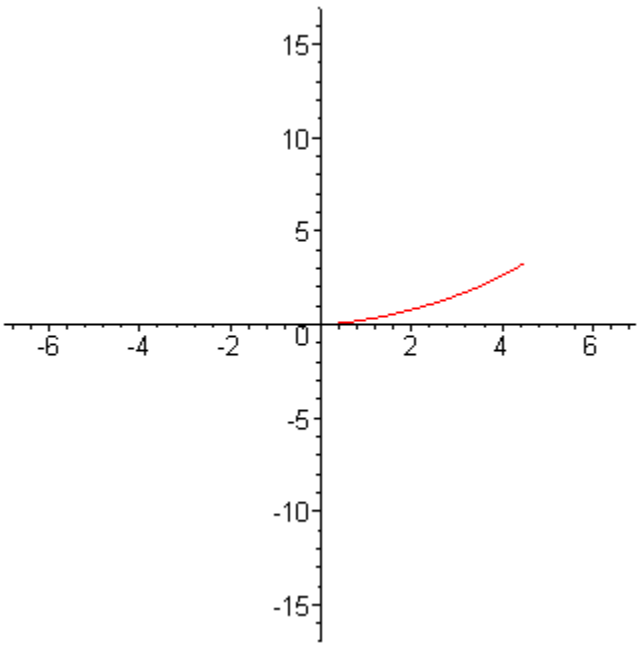
The following set of plots shows the progression of the skateboarder.

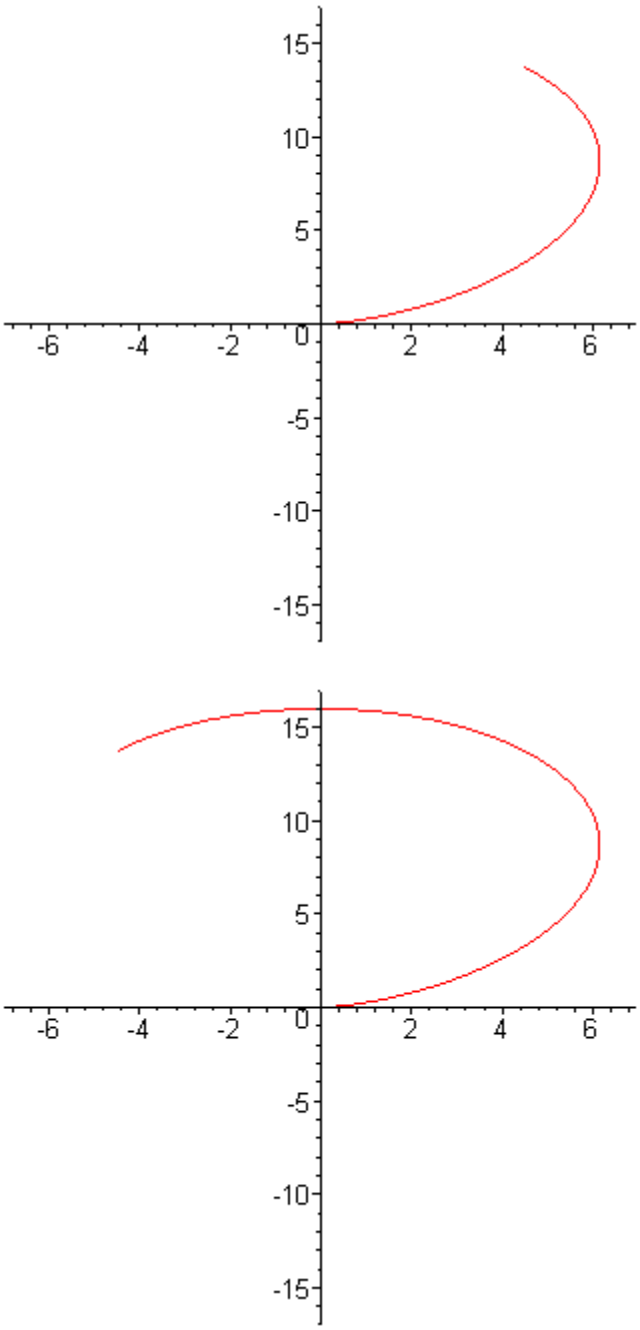
```
> with(plots):
  with(plottools):
```

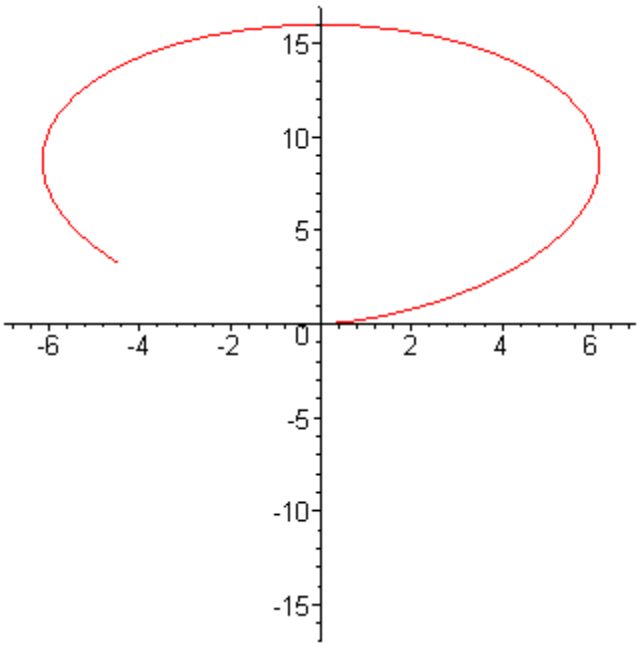
Warning, the name changecoords has been redefined

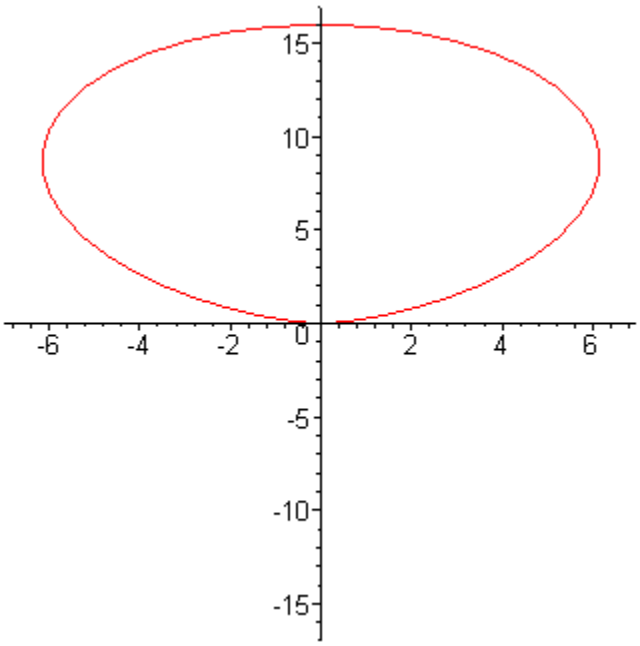
Warning, the assigned name arrow now has a global binding

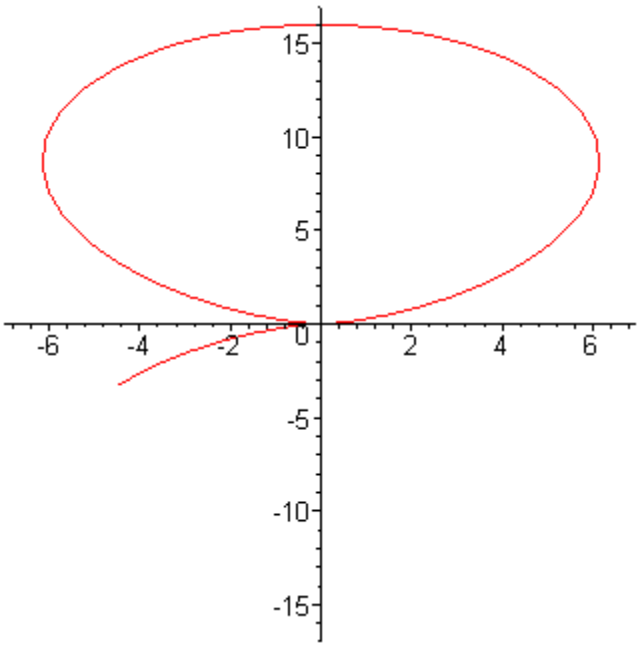
```
> for n from 0.1 to 1  by 0.1 do
    plot(r(t),t=0...2*Pi*n, view=[-7..7,-17..17],  coords=polar);
  od;
```

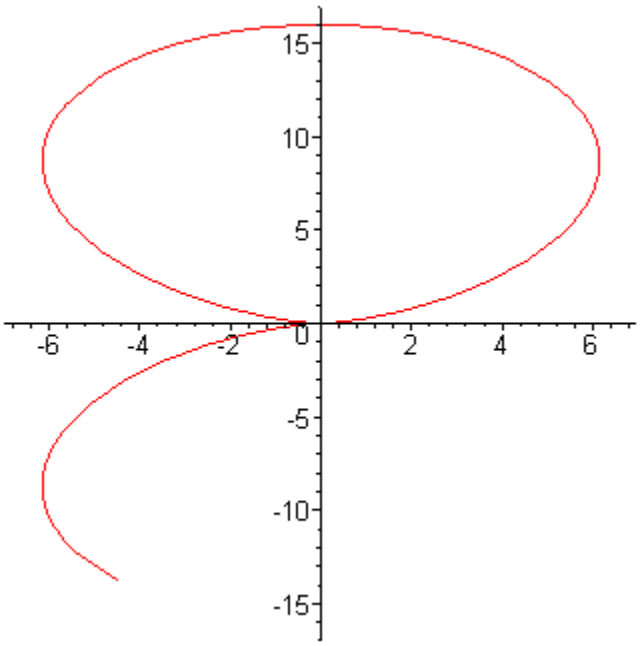


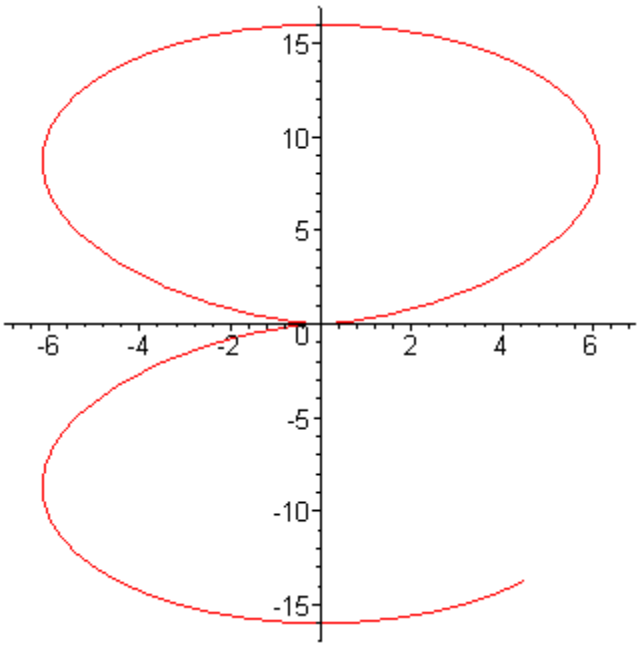

15
10
5
-6
-4
-2
0
2
4
6
-5
-10
-15
15
10
5
-6
-4
-2
0
2
4
6
-5
-10
-15

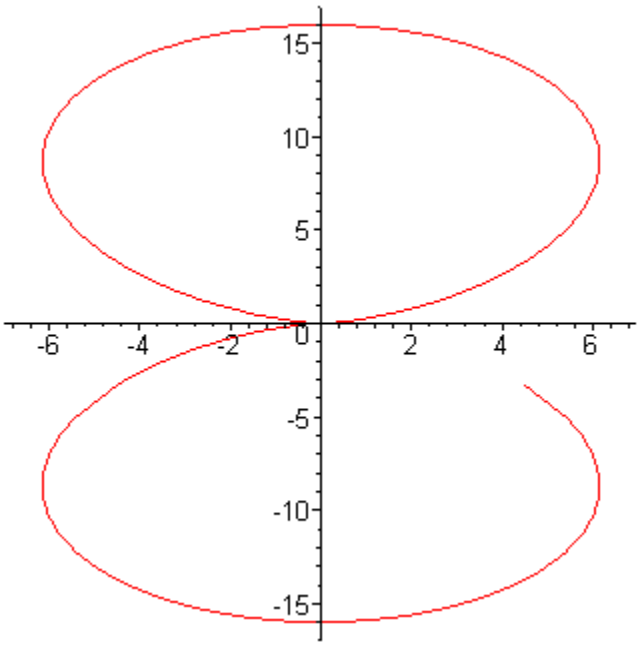

15
10
5
-6
-4
-2
0
2
4
6
-5
-10
-15

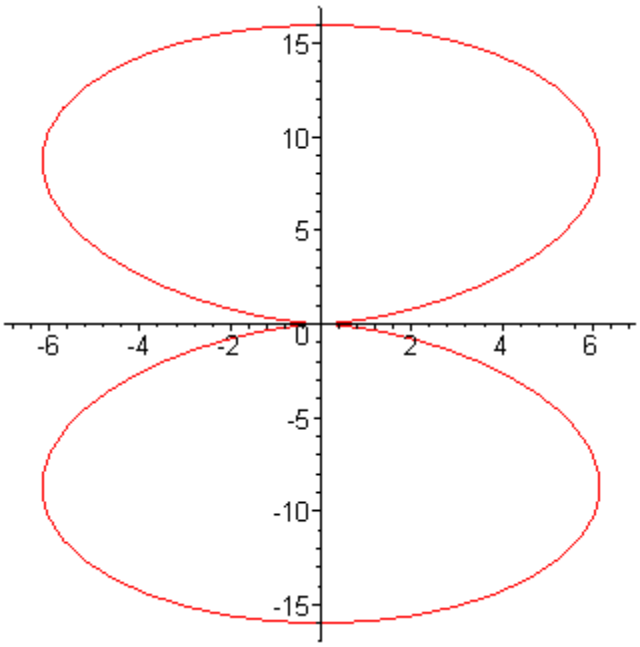

15
10
5
-6
-4
-2
0
2
4
6
-5
-10
-15


15
10
5
-6
-4
-2
0
2
4
6
-5
-10
-15


15
10
5
-6
-4
-2
0
2
4
6
-5
-10
-15


15
10
5
-6
-4
-2
0
2
4
6
-5
-10
-15


15
10
5
-6
-4
-2
0
2
4
6
-5
-10
-15



Now we animate the sequence. Click on the plot below to bring up the animation toolbar. Click on the **PLAY** button to view the animation. You can speed up or slow down the animation by clicking the appropriate button.

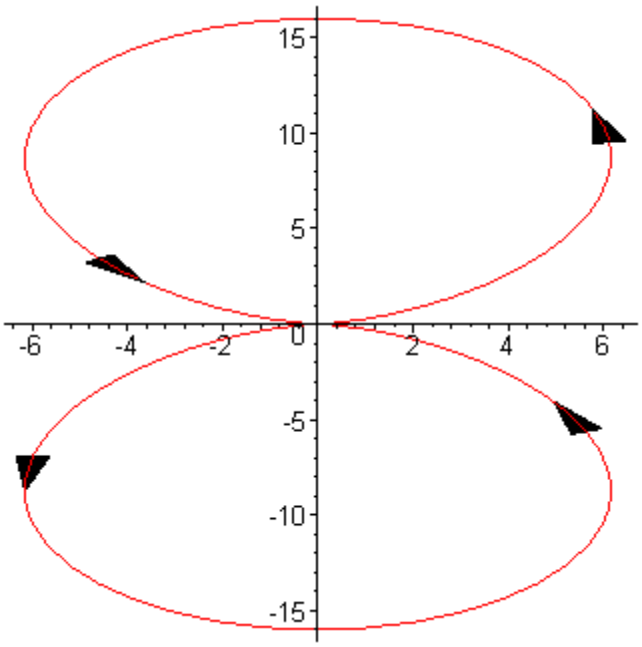
When we introduce vector quantities such as velocity and gradient, it is convenient to use rectangular coordinates. For this reason we will redefine our motion and position, velocity, and unit tangent vectors parametrically in x and y, using the above parametrizations for r and θ .

In order to show the direction of movement along the figure-8, we will place a few arrows on the graph.

```
> x1:=t->r(t)*cos(t);
  y1:=t->r(t)*sin(t);
  pp8:=plot([x1(t),y1(t),t=0..2*Pi]):
  A1:=arrow([x1(1.0), y1(1.0)], [x1(1.1), y1(1.1)], .3,.7,1,color=black):
  A2:=arrow([x1(2.5), y1(2.5)], [x1(2.6), y1(2.6)], .3,.7,1,color=black):
  A3:=arrow([x1(4.0), y1(4.0)], [x1(4.1), y1(4.1)], .3,.7,1,color=black):
  A4:=arrow([x1(5.5), y1(5.5)], [x1(5.6), y1(5.6)], .3,.7,1,color=black):
  display({pp8,A1,A2,A3,A4});
```

$$x1 := t \rightarrow \mathrm{r}(t)\cos(t)$$

$$y1 := t \rightarrow \mathrm{r}(t)\sin(t)$$



In the following parts, we will need the unit tangent vector, so we define now the quantities we will use .

```
> position:=[x1(t),y1(t)];
  velocity:= diff(position,t);
  evalc(linalg[dotprod](velocity, velocity)):
  speed:=sqrt(%);
  unittangent:=simplify([velocity[1]/speed,velocity[2]/speed]);
```

$$position := [16\sin(t)^2\cos(t),\ 16\sin(t)^3]$$

$$velocity := [32\sin(t)\cos(t)^2 - 16\sin(t)^3,\ 48\sin(t)^2\cos(t)]$$

$$speed := \sqrt{(32\sin(t)\cos(t)^2 - 16\sin(t)^3)^2 + 2304\sin(t)^4\cos(t)^2}$$

$$unittangent := \left[\frac{(3\cos(t)^2 - 1)\sin(t)}{\sqrt{\sin(t)^2(3\cos(t)^2+1)}},\ \frac{3\sin(t)^2\cos(t)}{\sqrt{\sin(t)^2(3\cos(t)^2+1)}}\right]$$

You Try It: Part I

Select and plot your own parametric equations for *x* and *y*. Replace the *x2* and *y2* functions. Remember that you are laying out a path for a skateboarder. You can execute the following example if you wish, and then try one of your own.

```
> x2:=t->10*sin(t-1)^2:
```

```
y2:=t->10*cos(2*t)^3:
pos:=simplify([x2(t),y2(t)]);
vel:=diff(pos,t);
linalg[dotprod](vel, vel):
spd:=simplify(sqrt(%)):
unittan:=simplify(vel/spd);
```

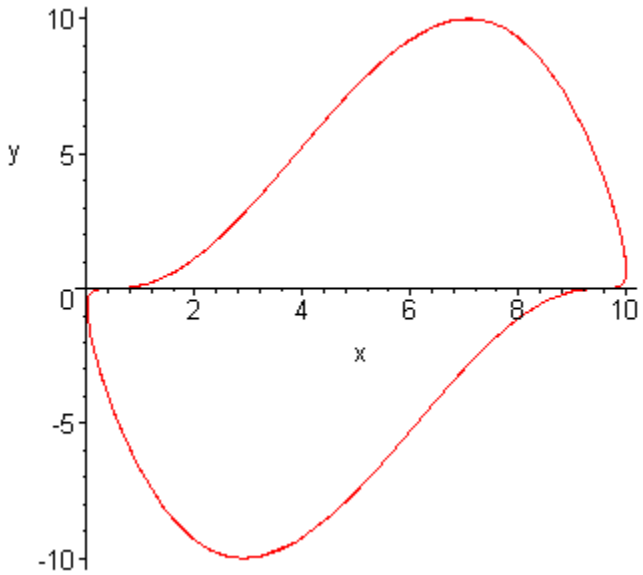
$$pos := [10 - 10\cos(t-1)^2,\ 10\cos(2t)^3]$$

$$vel := [20\cos(t-1)\sin(t-1),\ -60\cos(2t)^2\sin(2t)]$$

$$unittan := \frac{1}{20}\frac{[20\cos(t-1)\sin(t-1),\ -60\cos(2t)^2\sin(2t)]}{\sqrt{\left|-\cos(t-1)^2+\cos(t-1)^4\right|+9\left|\cos(2t)^4\sin(2t)^2\right|}}$$

Look at the graph, and adjust the limits on t if necessary.

```
> ppnew:=plot([x2(t), y2(t), t=0..7], color=red, labels=["x","y"]):
  print(ppnew);
```



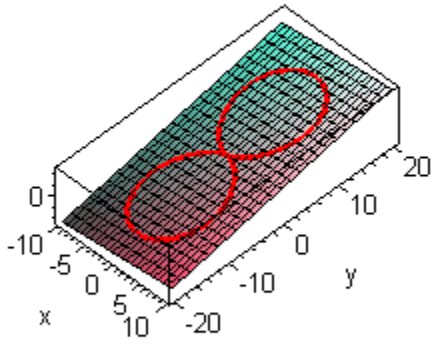
Part II: Skateboarding on a Planar Ramp and Computing and Analyzing the Directional Derivative

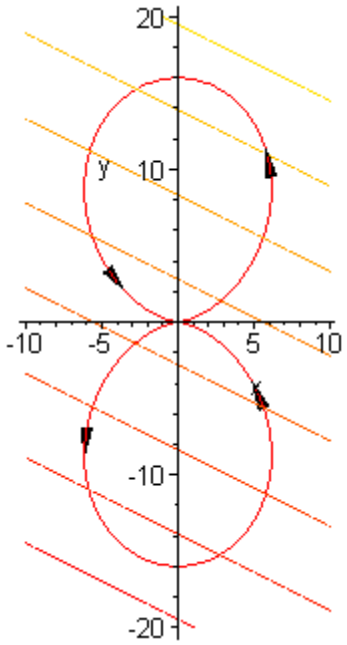
Now we consider a flat, inclined ramp for the skateboarder. The ramp has a 10% grade along the *x* direction (east-west) and a 20% grade along the *y* direction (north-south). Take the figure-8 from Part I as the horizontal projection of the skateboarder's path of motion on the inclined plane. If you were straight above the skateboarder and were looking down, the skateboarder would appear to

move along the figure-8 in Part I. You would not see the incline of the plane. To visualize the ramp, together with the figure 8, we produce plots, first in three-space, and then project the motion onto the horizontal *xy*-plane.

Click and drag anywhere on the 3D plot to view the graph from different angles.

```
> unassign('x','y');
  ramp:=(x,y)->.1*x+.2*y:
  p8ramp:=spacecurve([x1(t), y1(t), ramp(x1(t),y1(t)), t=0..2*Pi], color=red, thickness=3):
  pramp:=plot3d(ramp(x,y), x=-10..10, y=-20..20, axes=BOXED, orientation=[-45,45]):
  display({p8ramp,pramp}, scaling=constrained);
  ctramp:=contourplot(ramp(x,y), x=-10..10, y=-20..20):
  display({ctramp,pp8,A1,A2,A3,A4}, scaling=constrained);
```





Look at the contour plot above. Recall that the gradient of a function of two variables is a vector in the domain perpendicular to the contours. If you dot the gradient of the ramp into a unit vector in the direction of movement around the figure-8 in the plane, can you predict when that dot product will be 0 and where it is positive or negative?

Let's calculate the directional derivative of the height of the ramp relative to the changing positions along the skateboarder's path in the domain. Note that we are interested only in the gradient of the *z*-function along the path of the figure-8.

Before computing the directional derivative, look at both the two and three dimensional plots. When might you expect the directional derivative to be 0? Why? When might you expect it to be a maximum or a minimum? Jot down your estimates and see how they compare to the computed results that follow.

```
> gradient := eval( [diff(ramp(x,y),x), diff(ramp(x,y),y)], [x=x1(t), y=y1(t)] ):
  dirderiv:=[x1(gradient[1]), y1(gradient[2])]:
  ddramp:=evalc(linalg[dotprod](dirderiv, unittangent));
```

$$ddramp := \frac{0.1586707046\,(3\cos(t)^2 - 1)\sin(t)}{\sqrt{\sin(t)^2\,(3\cos(t)^2 + 1)}} + \frac{0.3763862280\,\sin(t)^2\cos(t)}{\sqrt{\sin(t)^2\,(3\cos(t)^2 + 1)}}$$

Look at your result carefully. What happens to the value of the directional derivative when *t* is a multiple of π ? Try different values of *t* in the section below.

```
> eval(ddramp, t=1*Pi);
```

Error, numeric exception: division by zero

Let's look at both the left-hand and the right-hand limits for the directional derivative as $t \to \pi$.

```
> llim:=limit(ddramp, t=0, right):
  rlim:=limit(ddramp, t=0, left):
  print(`The left - hand limit is`, llim);
  print(`The right - hand limit is`, rlim);
```

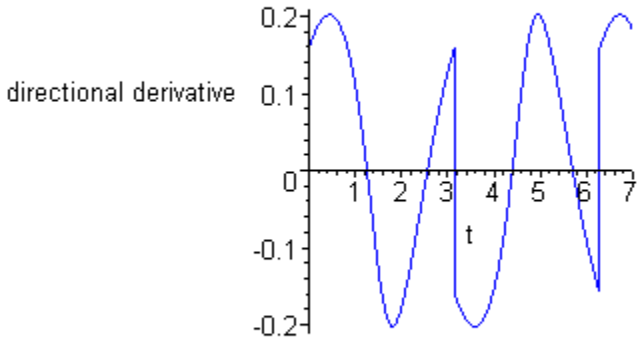
The left - hand limit is, 0.1586707046

The right - hand limit is, -0.1586707046

Uh-oh! What does this tell you about the value of the directional derivative at $t \to \pi$?

Now consider the plot of this directional derivative as the skateboarder traverses the entire figure-8.

```
> plot(ddramp, t=0..7, color=blue, labels=["t", "directional derivative"],tickmarks=[5,4]);
```



What do the sharp turns in this plot mean? Where are these sharp turns occurring? What has caused them? You may want to go back to your figure-8 graph and see what happens at $t = \pi$ and at $t = 2\pi$. The vertical segments drawn at $t = \pi$ and at $t = 2\pi$ don't really represent the directional derivative, because it actually takes a jump at those points. Is there any

way that you could adjust the path of the skateboarder to make the directional derivative continuous and the motion smooth at $t = \pi$ and at $t = 2\pi$?

What does it mean when the directional derivative is 0? We will encounter this later in the module on Lagrange multipliers. For later comparison, we go ahead and use the **solve( )** command to estimate the places where the directional derivative is 0, ignoring the places of discontinuity $t = \pi$ or 2π)

```
> dd1:=fsolve(ddramp,t=1.5):
  dd2:=fsolve(ddramp,t=2.8):
  dd3:=fsolve(ddramp,t=4.5):
  dd4:=fsolve(ddramp,t=6.0):
  ddzero:=[dd1,dd2,dd3,dd4]:
  print(`The directional derivative is zero for the following values of t:`, ddzero);
```

The directional derivative is zero for the following values of t:,
[1.252195378, 2.558446164, 4.393788032, 5.700038817]

```
> Digits:=8:
  seq([ddzero[i], x1(ddzero[i]), y1(ddzero[i]), ramp(x1(ddzero[i]),y1(ddzero[i]))], i=1..4):
  matrix([[`t`,`x`,`y`,`z`],%]);
```

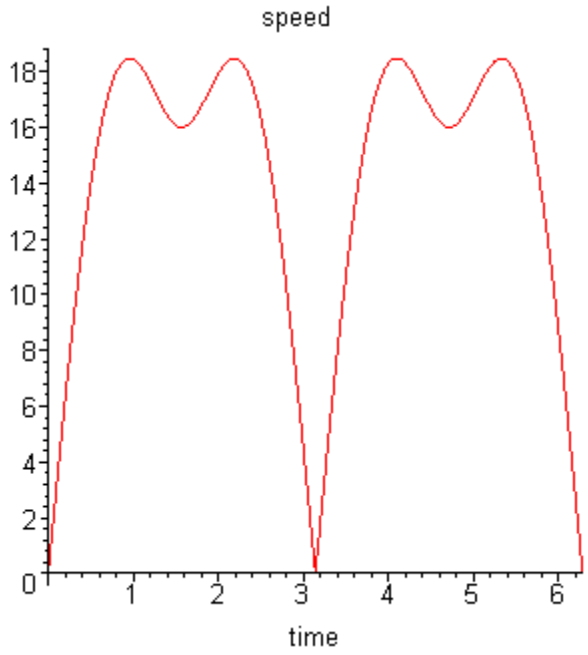
t	x	y	z
1.252195378	4.5200614	13.703908	3.1927877
2.558446164	-4.0497142	2.6714944	0.12932746
4.393788032	-4.5200621	-13.703908	-3.1927878
5.700038817	4.0497147	-2.6714951	-0.12932755

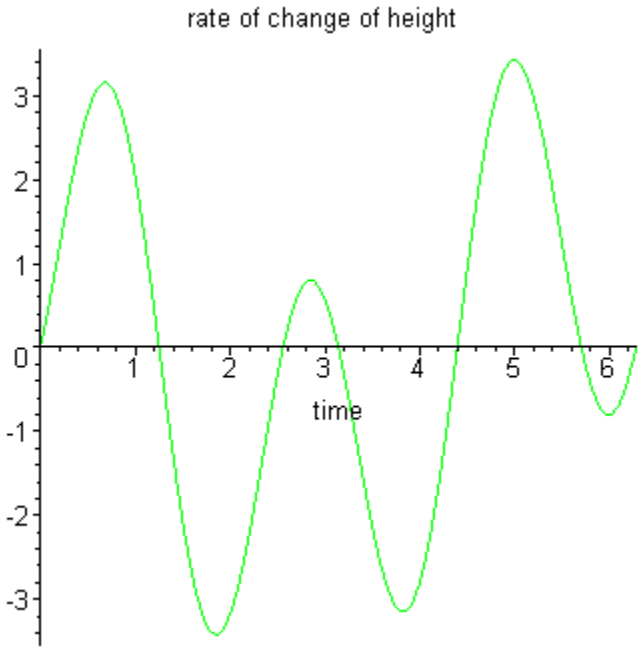
Where are these points on your plots?

Looking at the Time Derivative

If we assume that the parameter t represents time and determine the rate of change of the skateboarder's height with respect to time, we can multiply the directional derivative by the speed in the *xy*-plane. Why?

```
> plot(speed, t=0..2*Pi, labels=["time",""], title="speed");
  timederivramp:=ddramp*speed:
  print(``);
  plot(timederivramp, t=0..2*Pi, color=green, labels=["time",""],title= "rate of change
  height");
```





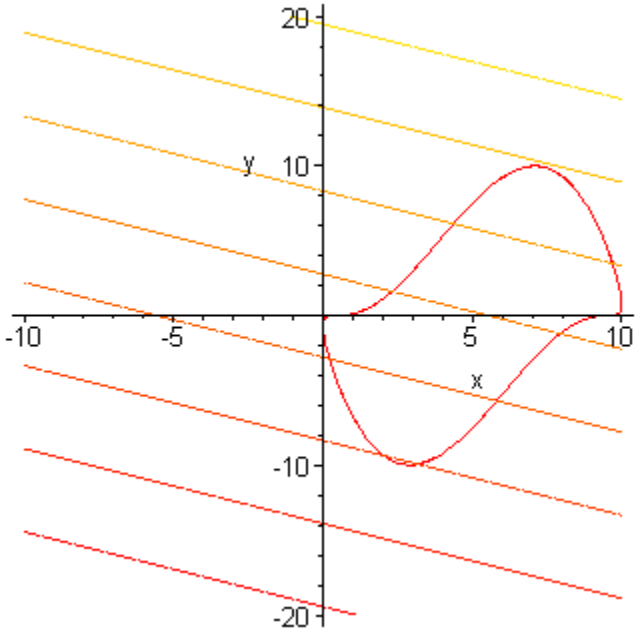
Interpret the speed plot. What is happening? Describe the skateboarder's motion. Why is the time derivative plot much smoother than the directional derivative?

You Try It: Part II

Take the curve that was defined in the You Try It: Part I, and picture it on this ramp.

> **x2:=t->10*sin(t-1)^2:**

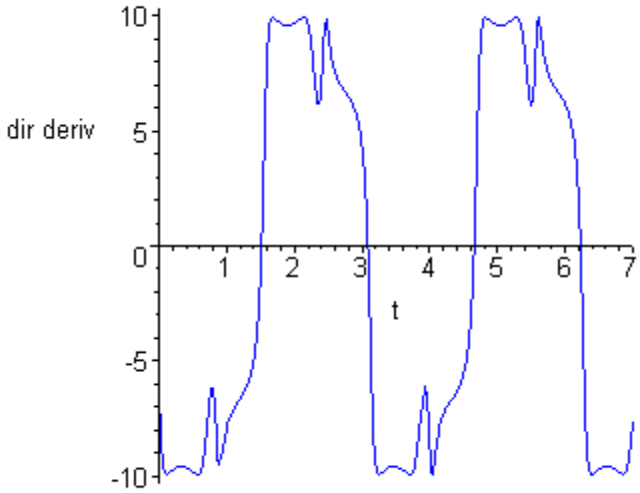
```
y2:=t->10*cos(2*t)^3:
pos:=simplify([x2(t),y2(t)]):
vel:=diff(pos,t):
linalg[dotprod](vel, vel):
spd:=simplify(sqrt(%)):
unittan:=simplify(vel/spd):
display({ppnew,ctramp});
```




```
> gradient := eval( [diff(ramp(x,y),x), diff(ramp(x,y),y)], [x=x2(t), y=y2(t)] ):
  dirderiv:= [x2(gradient[1]), y2(gradient[2])]:
  ddramp:=evalc(linalg[dotprod](dirderiv, unittan));
```

$$ddramp := \frac{6.1360105\cos(t-1)\sin(t-1)}{\sqrt{\left|-\cos(t-1)^2+\cos(t-1)^4\right|+9\cos(2t)^4\sin(2t)^2}} - \frac{23.441555\cos(2t)^2\sin(2t)}{\sqrt{\left|-\cos(t-1)^2+\cos(t-1)^4\right|+9\cos(2t)^4\sin(2t)^2}}$$

```
> plot(ddramp, t=0..7, color=blue, labels=[t,"dir deriv"]);
```

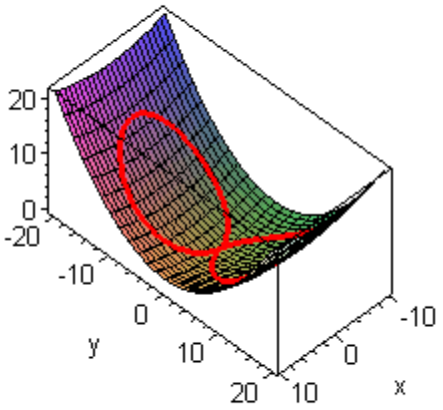


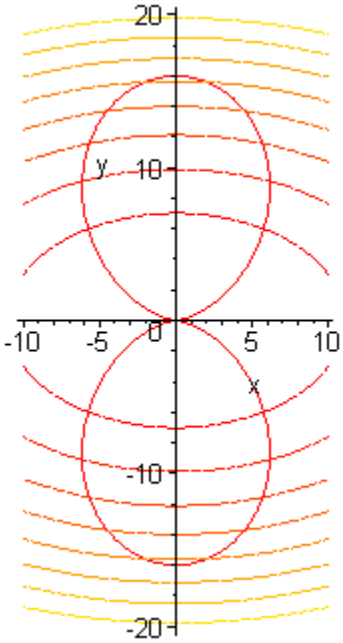
What is happening when you get sharp points in your directional derivative plot? What does it mena when the directional derivative is 0?

Part III: Skateboarding Inside an Elliptical Bowl

What if you were skateboarding on a bowl-shaped surface instead of on a planar ramp? Let's look at it graphically first.

```
> unassign('x','y');
  bowl:=(x,y)->.02*x^2+0.05*y^2:
  p8bowl:=spacecurve([x1(t), y1(t), bowl(x1(t),y1(t)), t=0..2*Pi], color=red, thickness=3):
  pbowl:=plot3d(bowl(x,y)-.5, x=-10..10, y=-20..20, axes=BOXED):
  print(display({p8bowl,pbowl}, scaling=constrained));
  ctbowl:=contourplot(bowl(x,y), x=-10..10, y=-20..20):
  print(display({ctbowl,pp8}, scaling=constrained));
```



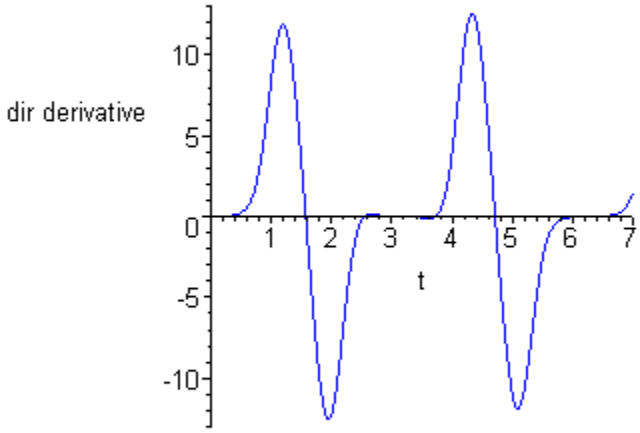


In case you are wondering why we subtracted .5 from the bowl height in the **plot3d** command, it was done for display purposes only, so that the figure-8 plot would show up better on top of the surface. As you did for the ramp, estimate when the directional derivative will be 0 or will reach its maximum or minimum, based on the dot product interpretation. Then, examine your directional derivative.

```
> gradient := eval( [diff(bowl(x,y),x), diff(bowl(x,y),y)], [x=x1(t), y=y1(t)] ):
  dirderiv:= [x1(gradient[1]), y1(gradient[2])]:
  ddbowl:=evalc(linalg[dotprod](dirderiv, unittangent));
```

$$ddbowl := \frac{16.\ \sin(0.64\ \sin(t)^2\cos(t))^2\ \cos(0.64\ \sin(t)^2\cos(t))\ (3\cos(t)^2-1)\ \sin(t)}{\sqrt{\sin(t)^2\ (3\cos(t)^2+1)}} + \frac{48.\ \sin(1.60\ \sin(t)^3)^3\ \sin(t)^2\cos(t)}{\sqrt{\sin(t)^2\ (3\cos(t)^2+1)}}$$

```
> plot(ddbowl, t=0..7, color=blue, labels=[t,"dir derivative"]);
```



As before, consider what it means when the directional derivative is 0? Also at what places on the figure 8 will the skateboarder be the highest?

```
> dd1:=fsolve(ddbowl, t=1.5 ):
  dd2:=fsolve(ddbowl, t=3.1 ):
  dd3:=fsolve(ddbowl, t=4.5):
  dd4:=fsolve(ddbowl, t=6.2 ):
  ddzero:=[dd1,dd2,dd3,dd4]:
  print(`The directional derviative is zero for the following values of t:`,ddzero);
```

The directional derviative is zero for the following values of t:, [1.5707963, 3.1414976, 4.712389(

```
> Digits:=8:
  seq([ddzero[i], x1(ddzero[i]), y1(ddzero[i]), bowl(x1(ddzero[i]),y1(ddzero[i]))], i=1..4):
  matrix([[`t`,`x`, `y`, `z`],%]);
```

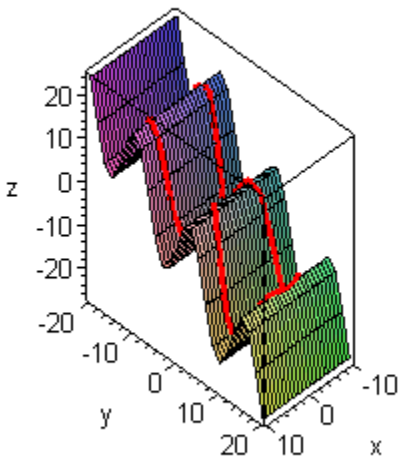
t	x	y	z
1.5707963	$0.42871834\ 10^{-6}$	16.000000	12.800000
3.1414976	$-0.14456296\ 10^{-6}$	$0.13741228\ 10^{-10}$	$0.41796899\ 10^{-15}$
4.7123890	$0.31384496\ 10^{-6}$	-16.000000	12.800000
6.2829742	$0.71305985\ 10^{-6}$	$-0.15053206\ 10^{-9}$	$0.10169088\ 10^{-13}$

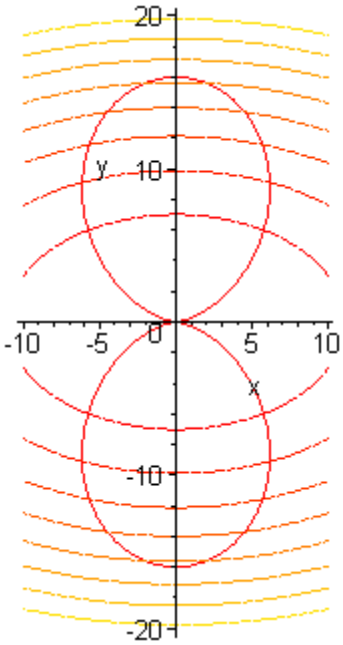
The directional derivative is 0 at each of the above points. Locate them on your surface and contour plots, and identify them as points of either maximum or minimum, or of neither.

You Try It: Part III

If you want to leave your path alone and just change your surface, alter the command in **rc** the following roller coaster type surface. This roller coaster gives a much more complex directional derivative for you to analyze.

```
> rc:=(x,y)->10*sin(y/2)-y:
  r:=t->16*sin(t)^2:
  theta:=t->t:
  parx:=t->r(t)*cos(theta(t)):
  pary:=t->r(t)*sin(theta(t)):
  p8rc:=spacecurve([parx(t), pary(t), rc(parx(t),pary(t)), t=0..2*Pi], color=red, thickness=3):
  prc:=plot3d(rc(x,y)-1, x=-10..10, y=-20..20, axes=BOXED, labels=[x,y,z]):
  print(display({p8rc,prc}, scaling=constrained));
  ctrc:=contourplot(rc(x,y), x=-10..10, y=-20..20):
  print(display({ctbowl,pp8}, scaling=constrained));
```

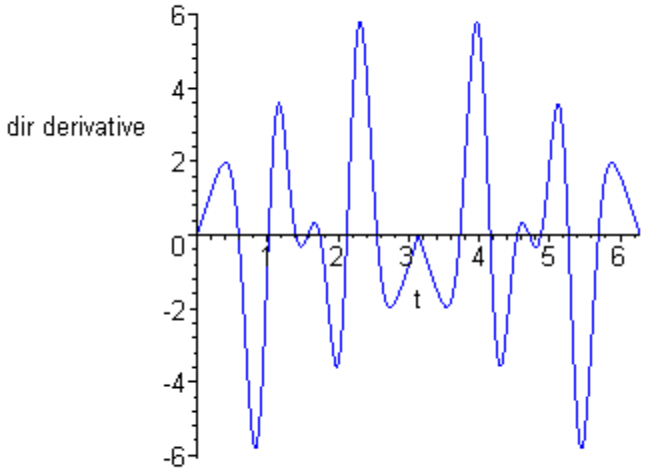




>

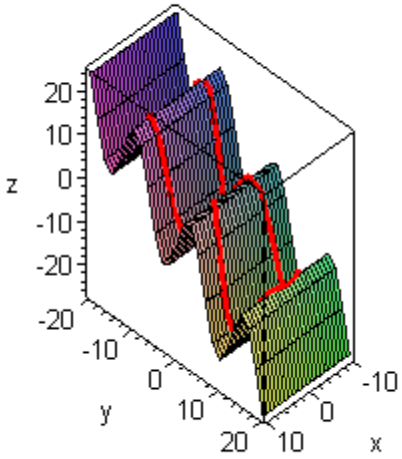
```
position:=[parx(t),pary(t)]:
velocity:= diff(position,t):
evalc(linalg[dotprod](velocity, velocity)):
speed:=sqrt(%):
unittangent:=simplify([velocity[1]/speed,velocity[2]/speed]):
gradient := eval( [diff(rc(x,y),x), diff(rc(x,y),y)], [x=parx(t), y=pary(t)] ):
ddrc:=(linalg[dotprod](gradient, unittangent));
plot(ddrc, t=0..2*Pi, color=blue, labels=["t","dir derivative"]);
```

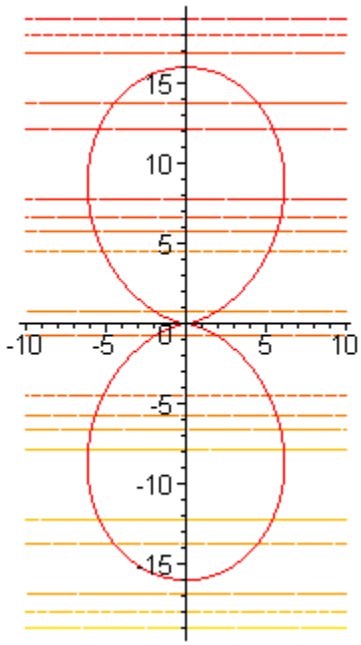
$$ddrc := 3\,(5\cos(8\sin(t)^3) - 1)\left(\frac{\sin(t)^2\cos(t)}{\sqrt{\sin(t)^2\,(3\cos(t)^2+1)}}\right)$$



If you want to change both your path and your surface, just remember that the domain for the surface has to be adjusted to contain the domain for the path. Alter the commands for the function **r** and **theta** in the first two lines and the function **rc**.

```
> r:=t->16*sin(t)^2:
  theta:=t->t:
  parx:=r(t)*cos(theta(t)):
  pary:=r(t)*sin(theta(t)):
  rc:=(x,y)->10*sin(y/2)-y:
  p8rc:=spacecurve([parx(t), pary(t), rc(parx(t),pary(t)), t=0..2*Pi], color=red, thickness=3):
  prc:=plot3d(rc(x,y)-1, x=-10..10, y=-20..20, axes=BOXED, labels=["x","y","z"]):
  print(display({p8rc,prc}, scaling=constrained));
  ctrc:=contourplot(rc(x,y), x=-10..10, y=-20..20):
  print(display({ctrc,pp8}, scaling=constrained));
```


z
20
10
0
-10
-20
-20
-10
0
10
20
y
10
0
-10
x


15
10
5
-10
-5
0
5
10
-5
-10
-15

>