Parametric and Polar Equations with a Figure Skater

Introduction

OBJECTIVE: Represent curves and analyze motion in parametric and polar form.

Parametric equations are very powerful and the purpose of this module is to help you get used to the idea of expressing curves using parametric equations and analyzing motion in the plane using these parametric equations. Polar plots can also be expressed parametrically, and that form can be translated easily into Cartesian coordinates.

▪ Technology Guidelines

NOTE: If you have just finished a module, restart *Mathematica* or close the *Kernel* before executing a new module.
TO OPEN CELLS, put your cursor on the right cell bracket and double click.
TO STOP AN EXECUTION
Select the *Kernel* pull-down menu and click on *Abort Evaluation.*
ORDER OF EXECUTION
Execute cells in the order given. Do not skip any Input cells within a given notebook.
SAVING NOTEBOOKS
You can save anytime to any directory you choose, and it is wise to save often.
However, before you do your final save, it is a good idea to delete all your output by selecting the
Delete All Output selection under the *Kernel* pull-down menu.
EXPERIENCING MAJOR PROBLEMS
Save if appropriate, then shut down *Mathematica* and start it up again.

Part I: Parametric Equations of a Curve in 2-Space

■ Defining the Function

First, we define the *x* and *y* coordinates parametrically. Suppose that time, *t*, is the independent variable. Once *x* and *y* are defined, we can write the position vector *r(t)*.

In[1]:=

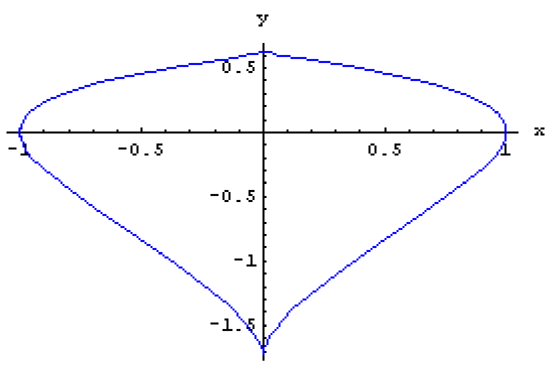
```
Off[General::spell]
Off[General::spell1]
Clear[x, y, t, r]
x[t_] := Cos[t]^3
y[t_] := 1 - Exp[Sin[t]]
 r[t_] := {x[t], y[t]}
Print["The position vector is ", r[t]]
```

The position vector is $\{\text{Cos}[t]^3, 1 - e^{\text{Sin}[t]}\}$

Now we plot the resulting curve in blue. In what direction are you moving on the curve as *t* increases?

In[8]:=

```
plotf = ParametricPlot[Evaluate[r[t]], {t, 0,
  PlotStyle -> RGBColor[0, 0, 1], AxesLabel ->

Print["The position vector is ", r[t]]
```



The position vector is $\{\text{Cos}[t]^3, 1 - e^{\text{Sin}[t]}\}$

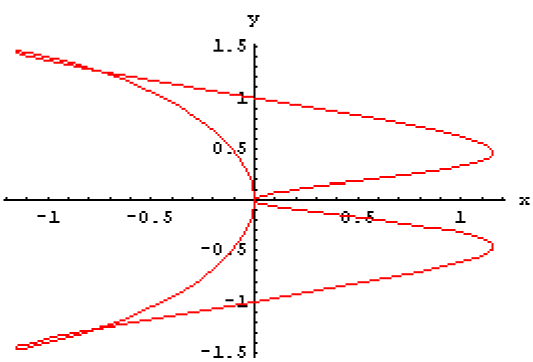
▪ Taking the Velocity and Acceleration into Account

If you consider the parametric equation as a vector equation for the motion of a particle, the velocity vector is found by differentiating each component of the position vector. Similarly, the acceleration vector is found by differentiating the components of the position vector twice. We do this with *Mathematica* and plot the velocity in red and the acceleration in green.

"> About *Mathematica*

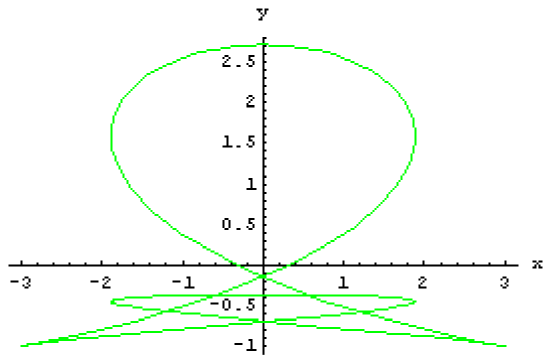
In[10]:=

```
plotfp = ParametricPlot[Evaluate[r'[t]], {t,
  PlotStyle -> RGBColor[1, 0, 0], AxesLabel ->

Print["The velocity function is ", r'[t]]

plotfpp = ParametricPlot[Evaluate[r''[t]], {t
  PlotStyle -> RGBColor[0, 1, 0], AxesLabel ->

Print["The acceleration function is  ", r''[
```



The velocity function is

$\{-3\,\text{Cos}[t]^2\,\text{Sin}[t],\ -e^{\text{Sin}[t]}\,\text{Cos}[t]\}$



The acceleration function is

$$\left\{-\frac{3}{4}\,(\text{Cos}[t] + 3\,\text{Cos}[3\,t]),\ e^{\text{Sin}[t]}\,(-\text{Cos}[t]^2 + \text{Sin}[t])\right\}$$

Note that the components of the velocity and acceleration functions are more complicated than the components of the postion function.

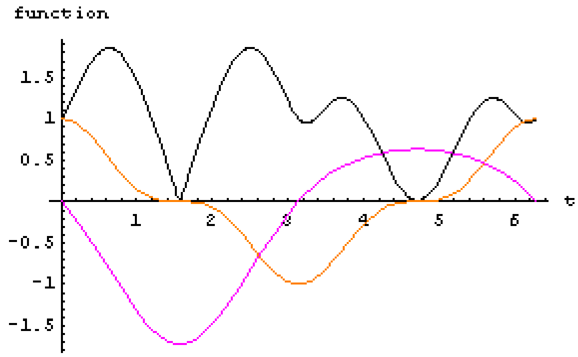
Let's look at the speed function and see what it tells us. The following plot shows the speed in black, the x-coordinate in orange and the y-coordinate in violet. Contrasting that to your parametric plot, identify the places where the speed function is 0.

In[14]:=

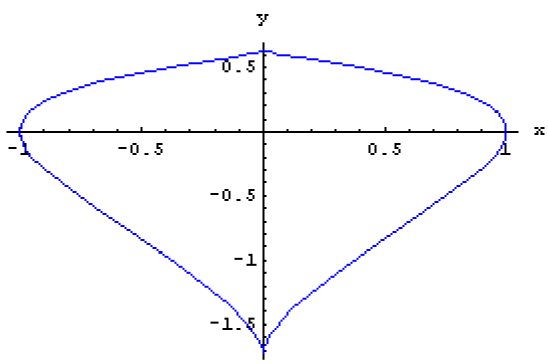
```
speed[t_] := √(r'[t].r'[t])

Print["The speed is ", speed[t]]

Plot[{speed[t], x[t], y[t]}, {t, 0, N[2 π]},
 PlotStyle -> {RGBColor[0, 0, 0], RGBColor[1,
   RGBColor[1, 0, 1]}, AxesLabel -> {t, functi

Print[
 "The speed is in black; the x-coordinate o
   of motion is in orange and the y-coordina
   path of motion is in violet."]

Show[plotf];
```

The speed is

$$\sqrt{e^{2\,\text{Sin}[t]}\,\text{Cos}[t]^2 + 9\,\text{Cos}[t]^4\,\text{Sin}[t]^2}$$




```
The speed is in black; the x-
  coordinate of the path of motion
  is in orange and the y-coordinate
  of the path of motion is in violet.
```



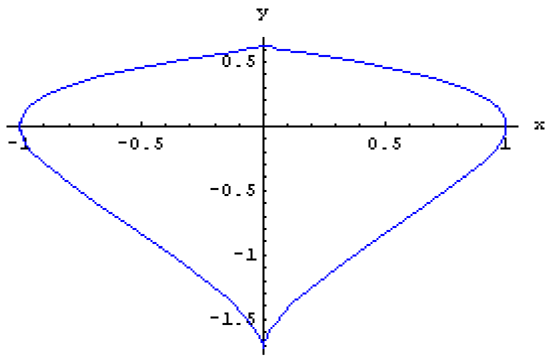
Identify the places on your path where the speed is 0 and the speed is a maximum.

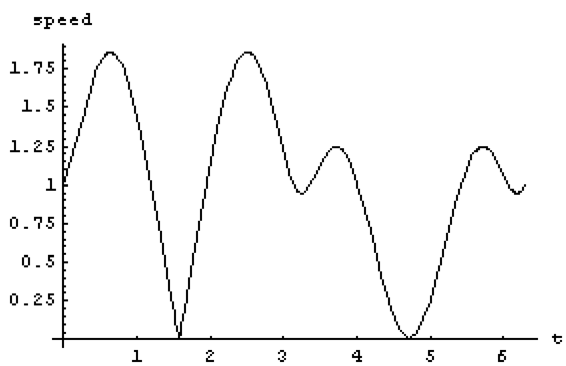
▪ Computing the Distance Traveled on a Curved Path

Suppose that you are walking along the path given above. The distance traveled can be found by integrating the speed function over a particular interval.

In[19]:=

```
Show[plotf];

Plot[speed[t], {t, 0, N[2 π]}, AxesLabel -> {t,

Print["The distance traveled around the clos
 distance = NIntegrate[speed[t], {t, 0, N[2 π]
```






```
The distance traveled around
   the closed path is 6.53062 units.
```

Think of this answer as either the distance around the curve or as the area under the speed function over the interval t from 0 to 2π.

You Try It: Part I

Defining and Plotting a Function

Select your own functions for **x[t_]** and for **y[t_]** in the cell below and then execute the command below. You need not select a closed path and you may wish to change the bounds for the parameter to something other than {t, 0 ,10}. Change the terms in red.

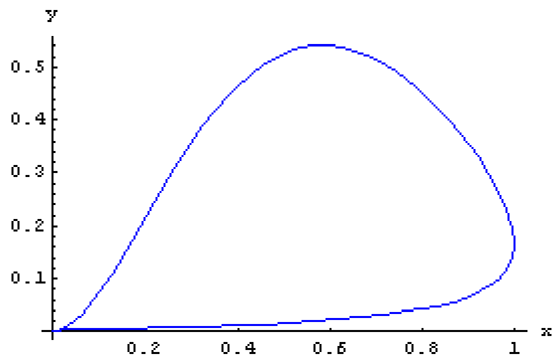
In[22]:=

```
Clear[x, y, t, r, speed]

x[t_] := Sin[t / 3.2]
```

```
y[t_] := Exp[-t] t^2

r[t_] = {x[t], y[t]}

plotf = ParametricPlot[Evaluate[r[t]], {t, 0,
  PlotStyle -> RGBColor[0, 0, 1], AxesLabel ->
```

Out[25]=

```
{Sin[0.3125 t], e^-t t^2}
```



- ## Computing the Velocity and Acceleration Vectors and Analyzing the Speed

In[27]:=

```
Print["The position function is ", r[t]]

Print["The velocity function is ", r'[t]]

Print["The acceleration function is  ", r''[

Print["The speed is ", speed[t_] = √(r'[t].r'

Plot[{speed[t], x[t], y[t]}, {t, 0, 10},
 PlotStyle -> {RGBColor[0, 0, 0], RGBColor[1,
   RGBColor[1, 0, 1]}, AxesLabel -> {t, functi
Print[
 "The speed is in black; the x-coordinate of
   of motion is in orange and the y-coordina
   path of motion is in violet."]
```

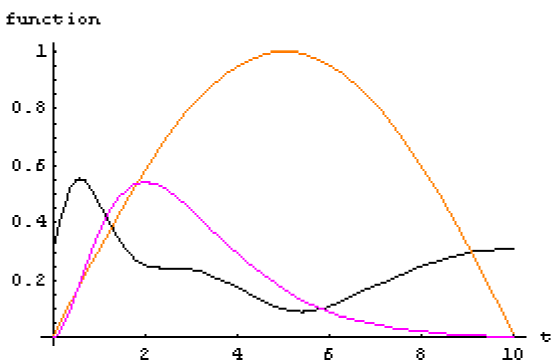
Show[plotf];

The position function is
$\{\mathrm{Sin}[0.3125\,t],\ e^{-t}\,t^2\}$

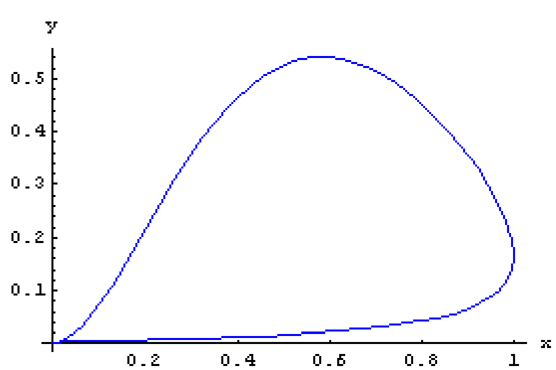
The velocity function is
$\{0.3125\,\mathrm{Cos}[0.3125\,t],\ 2\,e^{-t}\,t - e^{-t}\,t^2\}$

The acceleration function is
$\{-0.0976563\,\mathrm{Sin}[0.3125\,t],\ e^{-t}\,(2 - 4\,t + t^2)\}$

The speed is
$\sqrt{(2\,e^{-t}\,t - e^{-t}\,t^2)^2 + 0.0976563\,\mathrm{Cos}[0.3125\,t]^2}$



The speed is in black; the x-coordinate of the path of motion is in orange and the y-coordinate of the path of motion is in violet.



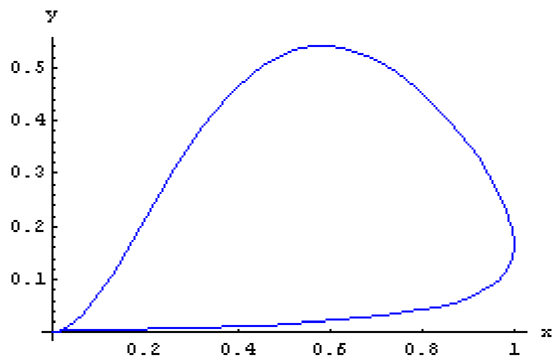
Identify the places on your path where the speed is 0 and the speed is a maximum.

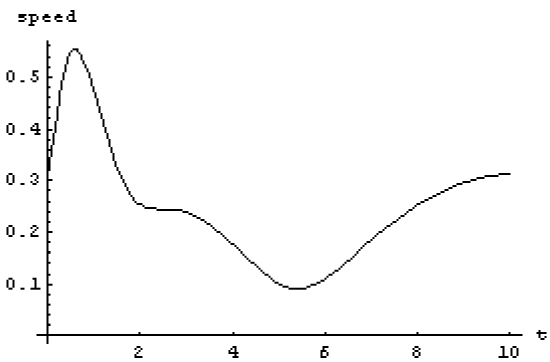
▪ Finding the Distance Along the Curved Path

Adjust the values of the parameter in the integration if you did so earlier.

In[33]:=

```
Show[plotf];
Plot[speed[t], {t, 0, 10}, AxesLabel -> {t, spe
Print["The distance traveled along the path
 distance = NIntegrate[speed[t], {t, 0, 10}],
```






```
The distance traveled along the path is
 2.46223 units.
```

Think of this answer as either the distance around the curve or as the area under the speed function.

Part II: A Figure Skater Tracing a Polar Plot

Four-Petal Pattern

Think of a figure skater tracing out a four-petal flower on the ice. The first set of commands gives the parametric equations of the figure skater in terms of a path that would be traced in the x-y plane. It is easiest to start with the equations in polar form.

In[34]:=

```
Off[General::spell]

Off[General::spell1]

Clear[r, θ, t, x, y]

r[t_] := 16 Sin[t]^2

θ[t_] := t / 2
```

You need to load a special graphics package to do polar plots. Execute this cell only once BEFORE you try to do a polar plot.
To write *r* as a function of θ, we had to replace the *t* in the r[t] with *t* as a function of θ. Since θ = t/2, t = 2θ.

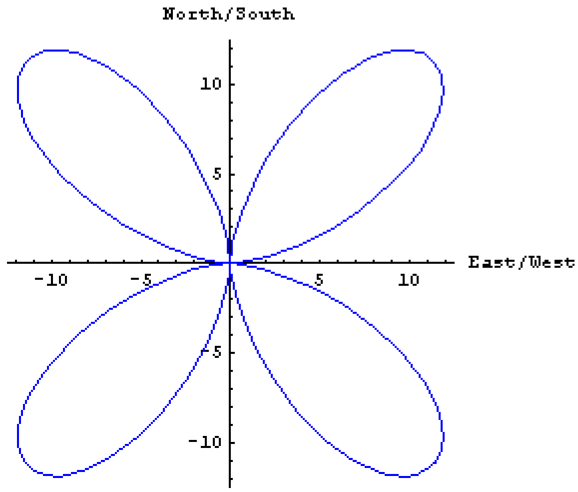
"> About *Mathematica*

In[39]:=

```
<< Graphics`Graphics`
```

In[40]:=

```
pp1 = PolarPlot[r[2 θ], {θ, 0, N[2 π]}, AspectR
  AxesLabel → {"East/West", "North/South"},
  PlotStyle -> RGBColor[0, 0, 1]];
```



For ease in computing equations of motion, we will define our position and velocity vectors parametrically in *x* and *y*, using the above parameterizations for *r* and θ. To show the direction of movement along the flower petals, we will place a few arrows on the graph (using the Epilog option). That requires a graphics arrow package. You must load the package before you execute a command within the package.

In[41]:=

```
<< Graphics`Arrow`
```

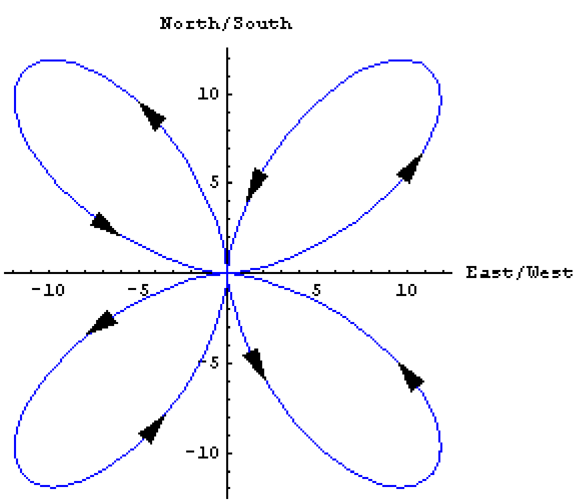
In[42]:=

```
Clear[velocity, speed]

parx[t_] = r[t] Cos[θ[t]];

pary[t_] = r[t] Sin[θ[t]];

position[t_] = {parx[t], pary[t]} // Simplify;

velocity[t_] = position'[t] // Simplify;

speed[t_] = √(velocity[t].velocity[t]) // Simp

Print["position vector = ", position[t]]

Print["speed = ", speed[t]]
```

```
pp4 = ParametricPlot[{parx[t], pary[t]}, {t, (
  AspectRatio -> Automatic, PlotStyle -> RGBCo
  AxesLabel -> {"East/West", "North/South"},
  Epilog →
   {Arrow[{parx[1], pary[1]}, {parx[1.1], pa
     HeadScaling → Absolute],
    Arrow[{parx[2.5], pary[2.5]}, {parx[2.6]
     HeadScaling → Absolute],
    Arrow[{parx[4], pary[4]}, {parx[4.1], pa
     HeadScaling → Absolute],
    Arrow[{parx[5.5], pary[5.5]}, {parx[5.6]
     HeadScaling → Absolute],
    Arrow[{parx[7], pary[7]}, {parx[7.1], pa
     HeadScaling → Absolute],
    Arrow[{parx[8.5], pary[8.5]}, {parx[8.6]
     HeadScaling → Absolute],
    Arrow[{parx[10], pary[10]}, {parx[10.1],
     HeadScaling → Absolute],
    Arrow[{parx[11.5], pary[11.5]}, {parx[11
     HeadScaling → Absolute]}];
```

position vector =
$\left\{16\,\mathrm{Cos}\left[\frac{t}{2}\right]\,\mathrm{Sin}[t]^2,\ 16\,\mathrm{Sin}\left[\frac{t}{2}\right]\,\mathrm{Sin}[t]^2\right\}$

speed = $4\sqrt{2}\,\sqrt{(17+15\,\mathrm{Cos}[2\,t])\,\mathrm{Sin}[t]^2}$

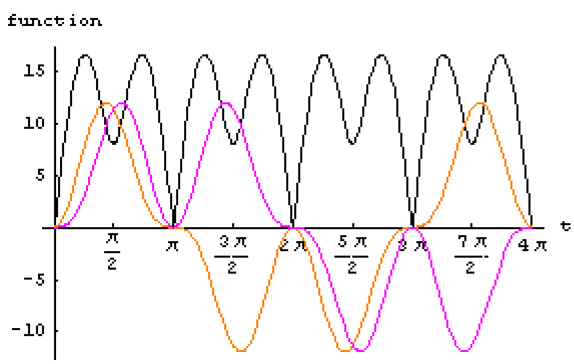


Can you tell in what direction the skater is moving just before and just after passing through at the origin?

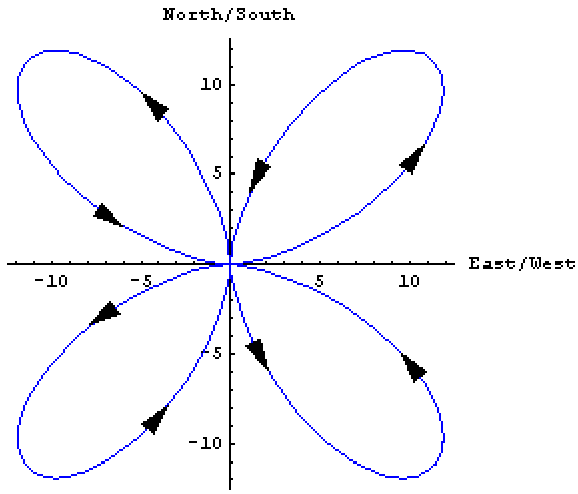
Let's look at the speed function and see what it tells us. The following plot shows the speed in black, the x-coordinate in orange and the y-coordinate in violet. Contrasting that to your parametric plot, identify the places where the speed function is 0.

In[51]:=

```
tickst = Table[n π / 2, {n, 0, 8}];

ticksf = Table[n, {n, -10, 15, 5}];

Plot[{speed[t], parx[t], pary[t]}, {t, 0, N[4
 PlotStyle -> {RGBColor[0, 0, 0], RGBColor[1,
   RGBColor[1, 0, 1]}, AxesLabel -> {t, functi
 Ticks -> {tickst, ticksf}];

Print[
 "The speed is in black; the x-coordinate o
   of motion is in orange and the y-coordin
   path of motion is in violet."]

Show[pp4];
```




```
The speed is in black; the x-
  coordinate of the path of motion
  is in orange and the y-coordinate
  of the path of motion is in violet.
```



Where are the places on your petal plot that the speed is zero?

▪ Velocity and Acceleration - When Are They Perpendicular?

Suppose we wish to determine for which values of *t* certain vectors describing the equations of motion are orthogonal. To do this, we will use the dot product, since perpendicular vectors yield a dot product of 0. Here we will examine when the velocity and acceleration vectors are perpendicular to one another. We begin by computing the dot product of the two vectors, and we plot the resulting function of *t* to get an idea of when the dot product might be 0.

In[56]:=

```
Clear[acceleration, vdota]

Print["velocity = ", velocity[t]]

Print["acceleration = ",
 acceleration[t_] = velocity'[t] // Simplify]

Print["velocity dotted into acceleration = "
 vdota = velocity[t].acceleration[t] // Simpli

Plot[Evaluate[vdota], {t, 0, N[4 π]}, AxesLab
        {t, "velocity dotted into accelerati
 Ticks -> {tickst, Automatic}];
```

velocity =

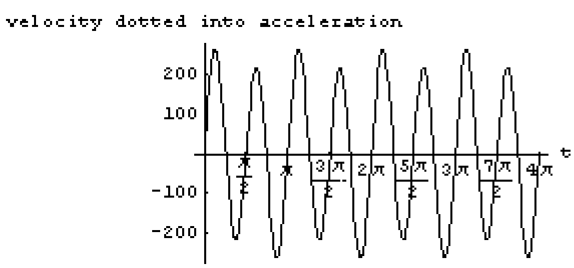
$$\left\{16\,\mathrm{Cos}\left[\frac{t}{2}\right]^2\,(-1+5\,\mathrm{Cos}[t])\,\mathrm{Sin}\left[\frac{t}{2}\right],\ 16\,\mathrm{Cos}\left[\frac{t}{2}\right]\,(1+5\,\mathrm{Cos}[t])\,\mathrm{Sin}\left[\frac{t}{2}\right]^2\right\}$$

acceleration =

$$\left\{2\,\mathrm{Cos}\left[\frac{t}{2}\right]\,(7-16\,\mathrm{Cos}[t]+25\,\mathrm{Cos}[2\,t]),\ 2\,(7+16\,\mathrm{Cos}[t]+25\,\mathrm{Cos}[2\,t])\,\mathrm{Sin}\left[\frac{t}{2}\right]\right\}$$

velocity dotted into acceleration =

$$32\,(17\,\mathrm{Cos}[t]+15\,\mathrm{Cos}[3\,t])\,\mathrm{Sin}[t]$$



As you can see from the graph, there are many times when the velocity and acceleration vectors are perpendicular to each other.

"> About *Mathematica*

The following commandsl find some of the places where the velocity and acceleration vectors are perpendicular. Note that we use seed values in **FindRoot** that seem close to some of the places where our function crosses the horizontal axis.

In[61]:=

```
Clear[vperpa]

vperpa[0] = FindRoot[vdota == 0, {t, 0}]

vperpa[1] = FindRoot[vdota == 0, {t, 1}]
```

```
vperpa[2] = FindRoot[vdota == 0, {t, 2}]

vperpa[3] = FindRoot[vdota == 0, {t, 3}]

vperpa[4] = FindRoot[vdota == 0, {t, 4}]

vperpa[5] = FindRoot[vdota == 0, {t, 5}]

vperpa[6] = FindRoot[vdota == 0, {t, 6}]

vperpa[7] = FindRoot[vdota == 0, {t, 7}]

vperpa[8] = FindRoot[vdota == 0, {t, 8}]

vperpa[9] = FindRoot[vdota == 0, {t, 9}]

vperpa[10] = FindRoot[vdota == 0, {t, 10}]
```

Out[62]=

$\{t \to 0.\}$

Out[63]=

$\{t \to 0.818756\}$

Out[64]=

$\{t \to 3.14159\}$

Out[65]=

$\{t \to 3.14159\}$

Out[66]=

$\{t \to 3.96035\}$

Out[67]=

$\{t \to 4.71239\}$

Out[68]=

$\{t \to 6.28319\}$

Out[69]=

```
{t → 7.10194}
```

Out[70]=

```
{t → 7.85398}
```

Out[71]=

```
{t → 5.46443}
```

Out[72]=

```
{t → 10.2435}
```

Now we evaluate *x* and *y* at the values of *t* we have found.

In[73]:=

```
special = Table[{parx[t], pary[t]} /. vperpa[i

TableForm[special, TableHeadings -> {None, {":
```

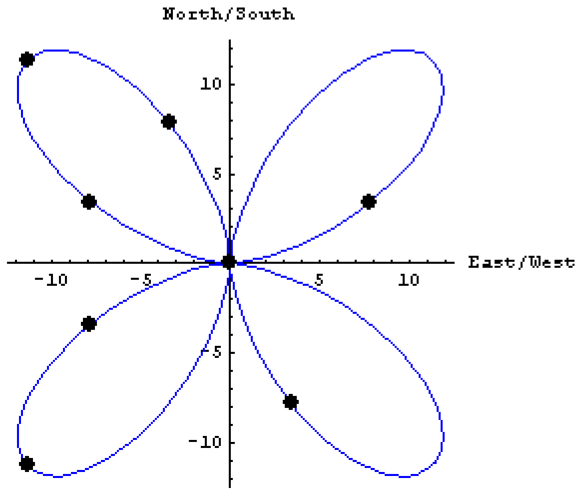
Out[74]//TableForm=

x	y
0.`	0.`
7.828211482499566`	3.3965987050341537`
1.4691952388169937`*^-47	2.39945715491542`*^-31
1.4691952388169937`*^-47	2.39945715491542`*^-31
-3.396598705034138`	7.8282114824995475`
-11.31370849898476`	11.313708498984761`
-9.59782861966168`*^-31	1.175356191053595`*^-46
-7.82821148249957`	-3.396598705034155`
-11.313708498984763`	-11.31370849898476`
-7.828211482499563`	3.39659870503415`
3.3965987050341626`	-7.8282114824995785`

We can now see where those points are relative to our petals.

In[75]:=

```
psp = ListPlot[special, PlotStyle -> PointSize
  DisplayFunction -> Identity];

Show[pp1, psp, DisplayFunction -> $DisplayFun
```



Can you see some points that we missed?

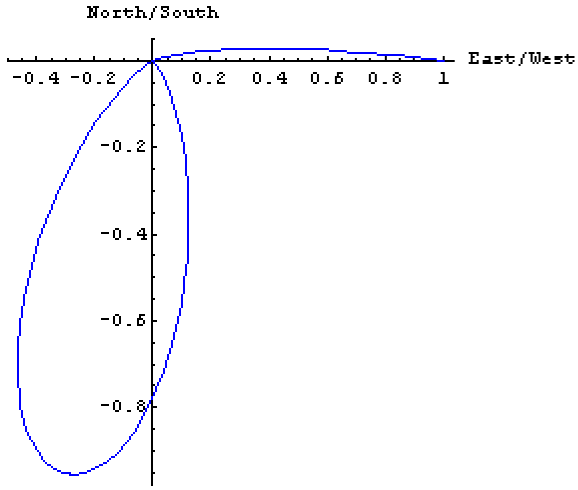
The dots show where the velocity and acceleration vectors are perpendicular to each other. Can you describe what is happening to the figure skater at those points?

You Try It: Part II

Try your own functions for *r[t]* and θ[t]. Remember to solve for *t* as a function of θ before you attempt to do a polar plot in the form of *r* as a function of θ. Replace the terms in red.

In[77]:=

```
Clear[r, θ, t, x, y]

r[t_] := Cos[t]^3

θ[t_] := t^2 / 8

pp1 = PolarPlot[Cos[√(8 θ)]^3, {θ, 0, 3}, Aspect
  AxesLabel → {"East/West", "North/South"},
  PlotStyle -> RGBColor[0, 0, 1]];
```



In[81]:=

```
Clear[velocity, speed]

parx[t_] = r[t] Cos[θ[t]];

pary[t_] = r[t] Sin[θ[t]];

position[t_] = {parx[t], pary[t]} // Simplify;

velocity[t_] = position'[t] // Simplify;

speed[t_] = √(velocity[t].velocity[t]) // Simp

Print["position vector = ", position[t]]

Print["speed = ", speed[t]]

Plot[{speed[t], parx[t], pary[t]}, {t, 0, 5},
 PlotStyle -> {RGBColor[0, 0, 0], RGBColor[1,
   RGBColor[1, 0, 1]}, PlotRange -> All,
 AxesLabel -> {t, function}];
```

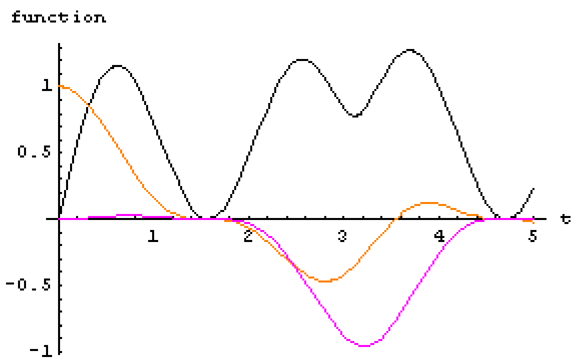
```
Print[
 "The speed is in black; the x-coordinate o
  of motion is in orange and the y-coordina
  path of motion is in violet."]

Show[pp1];
```

position vector =

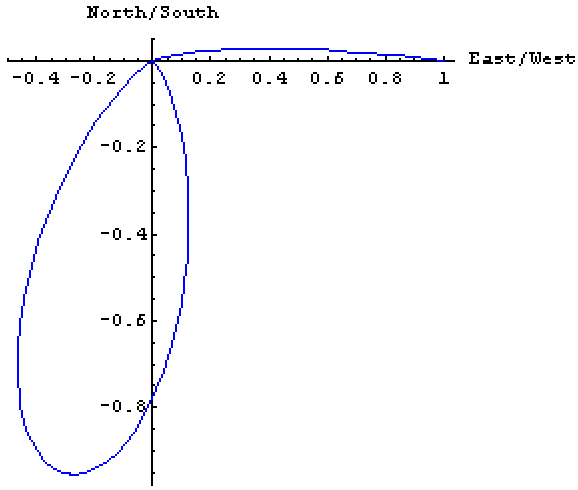
$$\left\{\mathrm{Cos}[t]^3\,\mathrm{Cos}\left[\frac{t^2}{8}\right],\ \mathrm{Cos}[t]^3\,\mathrm{Sin}\left[\frac{t^2}{8}\right]\right\}$$

speed =

$$\frac{\sqrt{\mathrm{Cos}[t]^4\,(144+t^2+(-144+t^2)\,\mathrm{Cos}[2\,t])}}{4\sqrt{2}}$$



The speed is in black; the x-
coordinate of the path of motion
is in orange and the y-coordinate
of the path of motion is in violet.



What is happening to your speed as *t* increases?

About *Mathematica*

In *Mathematica*, the derivative can be found with either the derivative command **D [function,variable to differentiate with respect to]** or the prime (single quote), **f'[x]**. For the prime to work, the function must be defined with an argument (**f[x]**), whereas the differentiation command can be applied to a function in any form, with or without an argument.
Go back.

If you try to execute the command before loading the package, *Mathematica* will be unable to execute the command, even if you try to load the package after you have already executed the command and then try re-executing the command. The best way to undo your mistake is to EXIT *Mathematica* completely and then start it again. Simply closing the notebook will not work; do save your notebook before you exit *Mathematica*.
Go back.

The **Solve[...]** command works sometimes, but usually only for algebraic (not transcendental) functions. The **FindRoot** command works most of the time. Using it is somewhat like using Newton's method, so you have to give it a starting point. Note that it yields approximate solutions and does not always converge to a solution.
Go back.