Curvas e Superfícies
▸ Lista 1
06/03/2024

Osmar Cardoso Lopes Filho
osmarclopesfilho@gmail.com

Esse documento possui códigos que geram animações no formato `gif` e que podem ser acessadas separadamente no repositório.

Questão 1

Queremos uma parametrização $\alpha(t) : I \to \mathbb{R}^2$ que descreva $x^2 + y^2 = 1$. O traço deve ser percorrido no sentido anti-horário e $\alpha(0) = (1, 0)$.

Primeiramente, escrevemos a representação implícita em coordenadas polares. Substituindo $x = \cos(\theta)r$ e $y = \sin(\theta)r$, obtemos o seguinte:

$$r(\cos^2(\theta) + \sin^2(\theta)) = 1$$
$$\Rightarrow \forall\theta \quad r = 1$$

Como r é contínuo, definimos θ como nosso único parâmetro, com intervalo $[0, 2\pi)$, obtendo, então, as seguintes equações:

$$x(\theta) = \cos(\theta)$$
$$y(\theta) = \sin(\theta)$$

Em seguida, escreveremos essas equações num ambiente CAS (*Computer Algebra System*) e computaremos o traço descrito por elas. O ambiente que usaremos será a linguagem de programação `Julia` equipada com os pacotes `Symbolics` e `Plots`. A seguir, temos o código que será usado:

```julia
using Symbolics
using Plots

@variables θ;

x = cos(θ);
y = sin(θ);

tlen = 100;
tvec = range(start=0, stop=2π, length=tlen);

# Initialize a plot with a single, empty series which
# the push! function inside the loop will target.
plt = plot(1; size = (600, 600), lim = (-1.2, 1.2), linewidth = 2);
anim = @animate for t in tvec
    xval = Symbolics.value(substitute(x, Dict(θ => t)));
    yval = Symbolics.value(substitute(y, Dict(θ => t)));

    push!(plt, (xval, yval));
end

gif(anim, "circle.gif");
```

A partir das equações, calculamos uma série de pontos a partir de entradas equidistantes no intervalo de θ. Esse calculo ocorre dentro de um `for` loop contido num macro `@animate`, o qual nos permite criar um `gif` do traço sendo criado. Os valores de x e y são calculados utilizando a função `substitute` do `Symbolics`, a qual troca θ pelas entradas que escolhemos em seu intervalo. O resultado é então convertido para um tipo padrão de `Julia` e é atribuido a sua variável. A seguir está o cículo criado:

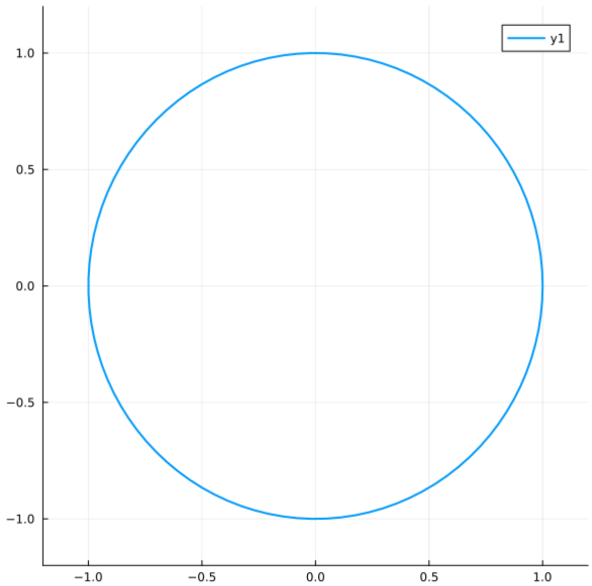

Figure 1: Circulo parametrizado. Curva animada

Em seguida, podemos calcular as derivadas dessas equações para construir vetores tangentes à curva. Fazemos isso continuando o código da seguinte forma:

```
Dθ = Differential(θ);

xDθ = expand_derivatives(Dθ(x));
yDθ = expand_derivatives(Dθ(y));

tanvec = [
    x        y
    x + xDθ  y + yDθ
];

v1 = Symbolics.value.(substitute.(tanvec, (Dict(θ => π/6),)));
v2 = Symbolics.value.(substitute.(tanvec, (Dict(θ => 2π/3),)));
v3 = Symbolics.value.(substitute.(tanvec, (Dict(θ => π),)));

# NaN entries are added to signal an interruption in the series, creating separated
lines.
v = [
    v1
    [NaN NaN]
    v2
    [NaN NaN]
    v3
];

plot!(plt, (v[:,1], v[:,2]); linecolor=:red, arrow=true)
```

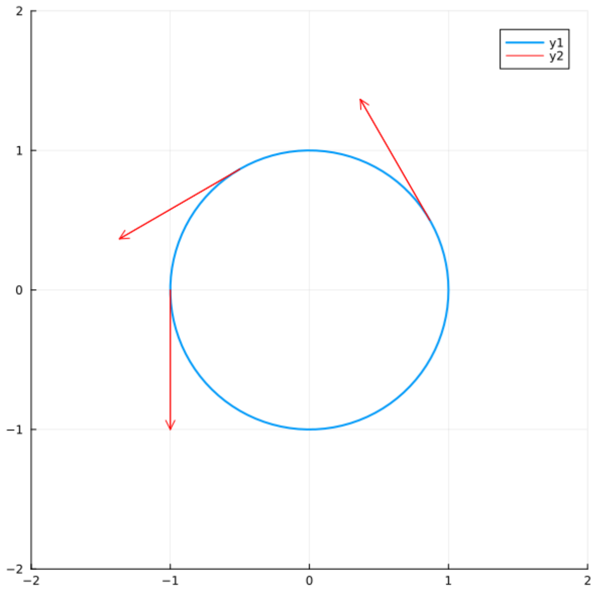

Figure 2: Circulo parametrizado com vetores tangentes. Curva e vetor animados

Questão 2

Queremos desenhar a *Limaçon*, curva parametrizada da seguinte forma:

$$\gamma(t) = ((1 + 2\cos(t))\cos(t), (1 + 2\cos(t))\sin(t)); t \in \mathbb{R}$$

Note que o código apresentado até o momento não é modularizado e falha em seguir uma das mais importantes regras idiomáticas de Julia: *Escreva funções, não apenas scripts*. Além de deixar o código mais legível e reutilizavel – ambas heurísticas para uma vida mais feliz – o compilador de Julia é melhor em otimizar funções do que código solto do escopo global.

Daqui em diante, começaremos a modularizar o código visto em questões passadas, reutilizando as função que criarmos. A seguir, está o código para desenhar a *limaçon*:

```
using Symbolics
using Plots

function curvepoints(xexp::Num, yexp::Num, param::Num, values)
    vlen = length(values)
    points = zeros((vlen, 2))

    for i = 1:vlen
        t = values[i]
        points[i, :] = Symbolics.value.(
          substitute.([xexp yexp], (Dict(param => t),))
        )
    end
    return points
end

@variables t

x = (1 + 2cos(t))*cos(t);
y = (1 + 2cos(t))*sin(t);
```

```
points = curvepoints(x, y, t, range(0, 2π, length = 120));

# For generating a still image
# plt = plot((points[:,1], points[:,2]); size=(500, 500), linewidth=2)

plt = plot(1; size=(500, 500), linewidth=2, xlim=(-0.5,3.2), ylim=(-2,2));
anim = @animate for i = 1:120
    push!(plt, (points[i,1], points[i,2]))
end

gif(anim, "pasnail.gif");
```

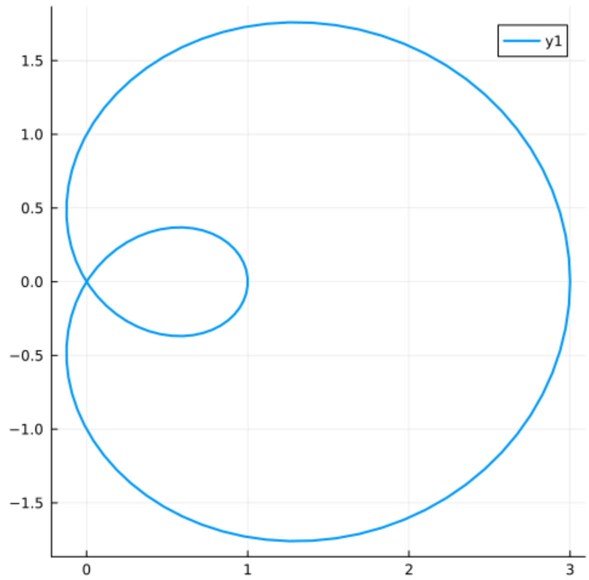

Figure 3: Limaçon. Curva animada

Essa curva passa duas vezes pelo ponto $(0, 0)$ no intervalo $[0, 2\pi)$. Visualizamos ambos os vetores tangentes a esse ponto ao acrescentar o seguinte código:

```
function tangvectors(xexp::Num, yexp::Num, param::Num, values)
    vlen = length(values)*3

    vectors = zeros((vlen, 2))

    D = Differential(param)
    dxexp = expand_derivatives(D(xexp))
    dyexp = expand_derivatives(D(yexp))

    tanvec = [
        xexp yexp (xexp + dxexp) (yexp +dyexp)
    ]

    for i = 1:3:vlen
        t = values[(i÷3) + 1]
        vectors[i, :] = Symbolics.value.(
            substitute.([xexp yexp], (Dict(param => t),))
        )
```

```
        vectors[i+1, :] = Symbolics.value.(
            substitute.([xexp+dxexp yexp+dyexp], (Dict(param => t),))
        )
        vectors[i+2, :] = [NaN NaN]
    end

    return vectors
end

vectors = tangvectors(x, y, t, [2π/3 4π/3]);
plot!(plt, (vectors[:,1], vectors[:,2]); linecolor=:red, arrow=true)
```

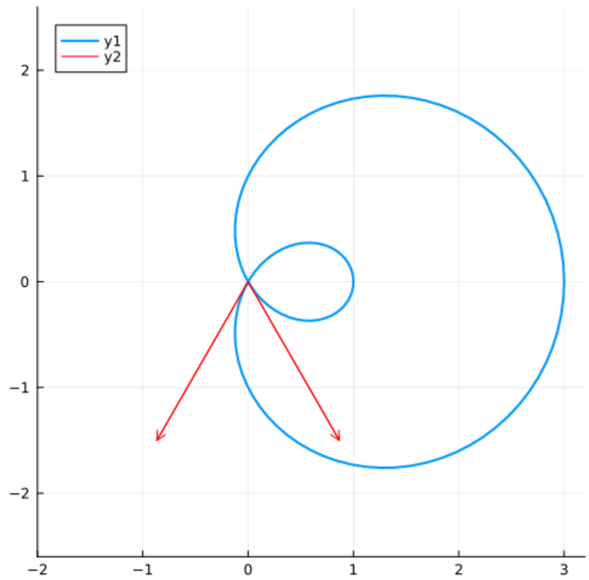

Figure 4: Limaçon com vetores tangentes à origem.

Questão 3

Queremos achar uma parametrização e desenhar a *Cissoide de Diocles*, curva definida implicitamente da seguinte forma:

$$x^3 + xy^2 - 2ay^2 = 0$$

Começamos nosso desenvolvimento isolando uma das variáveis:

$$\begin{aligned} x^3 + xy^2 - 2ay^2 &= 0 \\ \Rightarrow 1 + \frac{y^2}{x^2} - 2a\frac{y^2}{x^3} &= 0 \\ \Rightarrow \frac{y^2}{x^2}\left(1 - 2\frac{a}{x}\right) &= -1 \\ \Rightarrow \frac{y^2}{x^2} &= -\frac{1}{1 - 2\frac{a}{x}} \end{aligned}$$

Com uma única instância de x no lado direito, retiramos y da equação ao introduzir o parâmetro $t = \frac{y}{x}$:

$$\frac{y^2}{x^2} = -\frac{1}{1-2\frac{a}{x}} \Rightarrow t^2 = -\frac{1}{1-2\frac{a}{x}}$$
$$\Rightarrow t^2 - 2a\frac{t^2}{x} = -1 \Rightarrow t^2x - 2at^2 = -x$$
$$\Rightarrow x = \frac{2at^2}{t^2+1} \Rightarrow y = \frac{2at^3}{t^2+1}$$

Em seguida, desenhamos a curva:

```
@variables t a

x = (2*a*t^2)/(t^2 + 1);
y = (2*a*t^3)/(t^2 + 1);

x = substitute(x, Dict(a => 1));
y = substitute(y, Dict(a => 1));

points = curvepoints(x, y, t, range(-5, 5, 100));

# Generate still image
#plt = plot((points[:,1], points[:,2]); linewidth=2, size=(500, 500), xlim=(-5, 7),
ylim=(-6, 6));

plt = plot(1; linewidth=2, size=(500, 500), xlim=(-5, 7), ylim=(-6,6));
anim = @animate for i = 1:100
    push!(plt, points[i, 1], points[i, 2])
end

gif(anim, "diocis.gif"; fps=24);
```

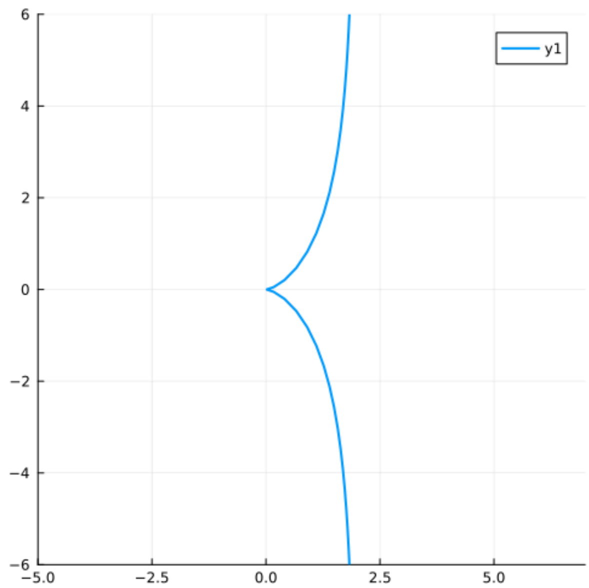

Figure 5: Cissoide de Diocles. Curva animada

A Cissoide foi usada numa das soluções para o problema de duplicar o cubo. Uma das soluções dos tempos antigos que não usava apenas régua e compasso.

Questão 4

Queremos parametrizar e desenhar o *Folium de Descartes*, curva definida implicitamente da seguinte forma:

$$x^3 + y^3 = 3xy$$

Começamos nosso desenvolvimento escrevendo a equação em coordenadas polares:

$$\begin{aligned} & r\left(\cos(\theta)^3 + \sin(\theta)^3\right) = 3\cos(\theta)\sin(\theta)r^2 \\ \Rightarrow\ & \cos(\theta)^3 + \sin(\theta)^3 = 3\cos(\theta)\sin(\theta)r \\ \Rightarrow\ & \frac{3\cos(\theta)\sin(\theta)}{\cos(\theta)^3 + \sin(\theta)^3} = r \end{aligned}$$

Concluimos com isso que podemos usar θ como nosso parâmetro. Desenhamos então a curva no intervalo $\left(-\pi\frac{1}{4}, \pi\frac{3}{4}\right)$:

```
using Symbolics
using Plots

function genvis(points; xlim=:auto, ylim=:auto)
    plt = plot(1; linewidth=2, size=(500,500), xlim=xlim, ylim=ylim)
    anim = @animate for i = 1:size(points, 1)
        push!(plt, points[i,1], points[i,2])
    end

    return (plt, anim)
end

# Multiple dispatch of genvis. We can either pass only points, or both points and
vectors.
# The compiler will figure out which version of the function to call.
function genvis(points, vectors; xlim=:auto, ylim=:auto)
    len = size(points, 1)
    anim = @animate for i = 1:len
        plot(points[1:i,1], points[1:i,2]; linewidth=2 size=(500,500), xlim=xlim,
ylim=ylim)
        plot!(vectors[1:i*3, 1], vectors[1:i*3, 2])
    end

      plt  =  plot(points[1:len,1],  points[1:len,2];  linewidth=2  size=(500,500),
xlim=xlim, ylim=ylim)

    return (plt, anim)
end

@variables θ

r = (3cos(θ)sin(θ))/(cos(θ)^3 + sin(θ)^3);

x = cos(θ)*r;
y = sin(θ)*r;

points = curvepoints(x, y, θ, range(-π/4 + 0.1, 3π/4 - 0.1, length=100));
```

```
vectors = tangvectors(x, y, θ, range(-π/4 + 0.1, 3π/4 - 0.1, length=100));
(plt, anim) = genvis(points, vectors; xlim=(-7,7), ylim=(-7,7));

gif(anim, "folidesc.gif"; fps=24)
plot(plt)
savefig(plt, "folidesc.png")
```

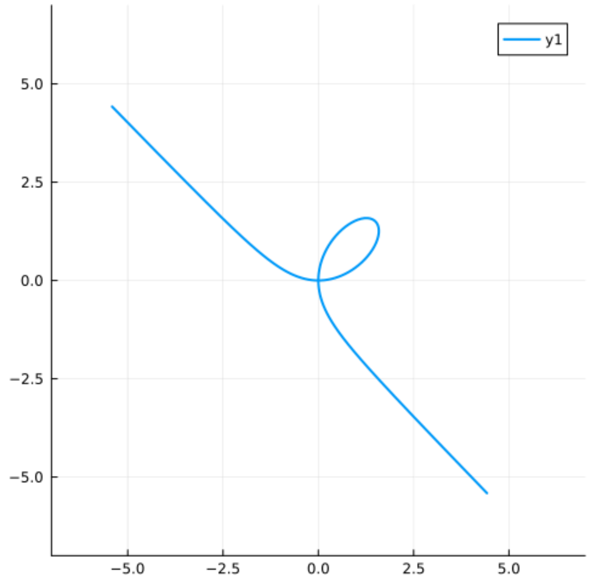

Figure 6: Folium de Descartes. Curva e vetor animados

Podemos também desenhar os contornos da equação implícita, mostrando as curvas geradas quando mudamos a igualdade a 0 para outros valores:

```
folidesc(x, y) = x^3 + y^3 - 3*x*y;
xvalues = range(-5, 5, length=150);
yvalues = range(-5, 5, length=150);
z = folidesc.(transpose(xvalues), yvalues);

levels = [
    range(-50, -5, step=5);
    range(-5, 5, step=0.5);
    range(5, 50, step=5);
];

con = contour(xvalues, yvalues, z; size=(500,500), levels=levels)
```

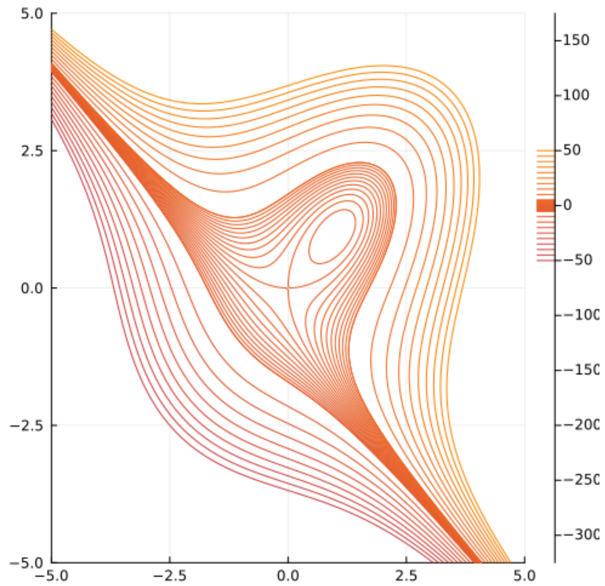

Figure 7: Curvas de nível do Folium

Questão 5

Queremos desenhar as seguintes parametrizações da parábola:

$$\alpha(t) = (t, t^2); \quad \gamma(t) = (t^3, t^6)$$

Fazemos isso da seguinte forma:

```
@variables t

x1 = t
y1 = t^2

x2 = t^3
y2 = t^6

points1 = curvepoints(x1, y1, t, range(-5, 5, length=100))
points2 = curvepoints(x2, y2, t, range(-2, 2, length=100))

vectors1 = tangvectors(x1, y1, t, range(-5, 5, length=100))
vectors2 = tangvectors(x2, y2, t, range(-2, 2, length=100))

(plt1, anim1) = genvis(points1, vectors1)
(plt2, anim2) = genvis(points2, vectors2)

savefig(plt1, "parabol1.png")
savefig(plt2, "parabol2.png")
gif(anim1, "parabol1.gif")
gif(anim2, "parabol2.gif")
```

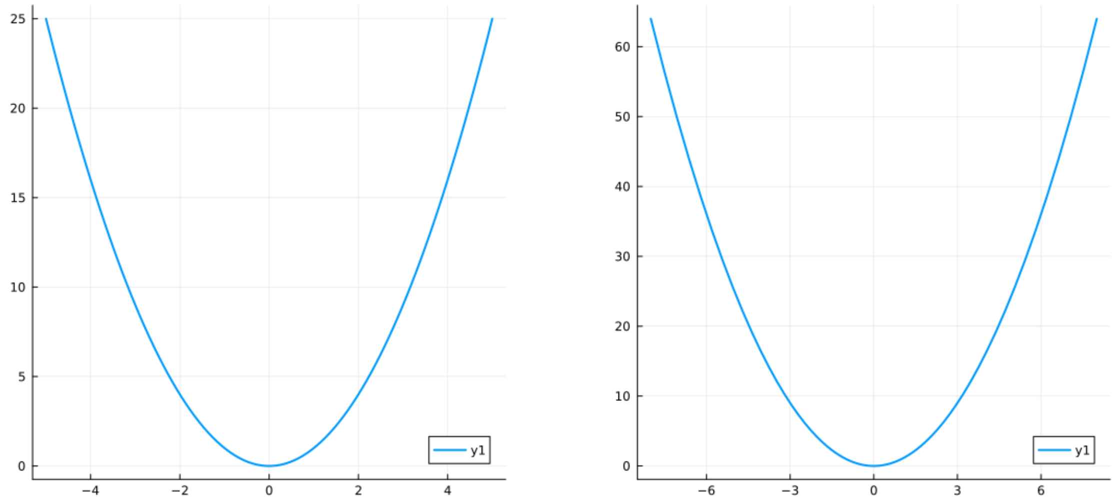

Table 1: Parábolas. Primeira parábola | Segunda parábola

A seguir desenhamos as derivadas dessas curvas. Será, então, fácil ver que a segunda curva não é regular, já que sua derivada é nula quando $t = 0$:

```
Dt = Differential(t);
(dx1, dy1, dx2, dy2) = expand_derivatives.(Dt.((x1, y1, x2, y2)));

dpts1 = curvepoints(dx1, dy1, t, range(-5, 5, length=100))
dpts2 = curvepoints(dx2, dy2, t, range(-2, 2, length=100))

(pltd1, _) = genvis(dpts1; xlim=(-2,4))
(pltd2, _) = genvis(dpts2; xlim=(-2,8))

savefig(pltd1, "derparab1.png")
savefig(pltd2, "derparab2.png")
```

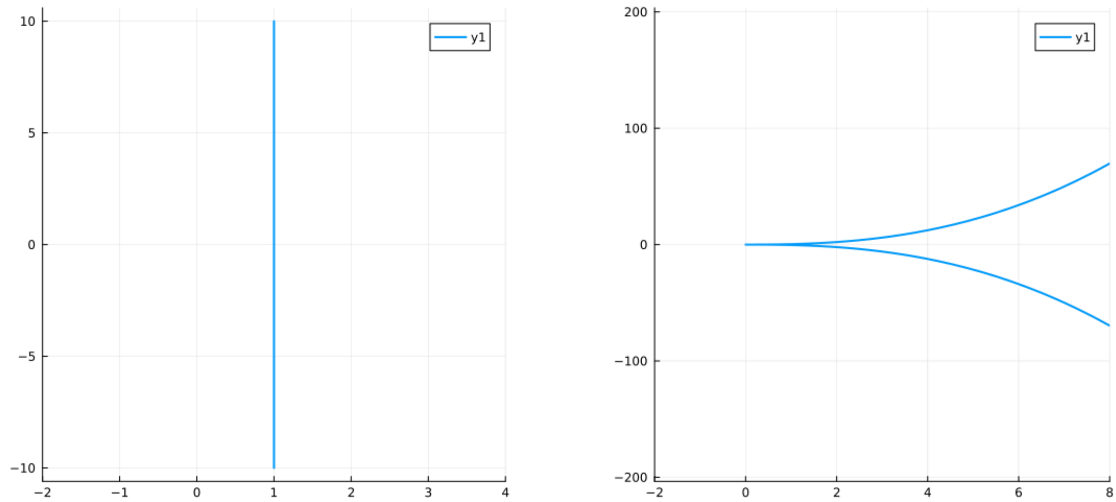

Table 2: Derivadas das parábolas

Uma tentativa intuitiva de reparametrização entre as curvas seria a aplicação da função $f(t) = t^2$ no intervalo da segunda curva. Os valores resultantes disso seriam usados na primeira curva e resultariam na mesma parábola. Contudo, essa função tem uma imagem contida nos positivos e, portanto, não cobre todo o intervalo da primeira equação.

Questão 6

Escolhemos a curva *hipotrocoide*, desenhada por um ponto fixo em referência a um círculo que gira no interior de outro centrado na origem.

As equações que regem o movimento desse ponto são as seguintes:

$$C = ((R - r)\cos(\theta), (R - r)\sin(\theta))$$
$$\varphi = \theta\left(\frac{R}{r} - 1\right)$$
$$P = C + (d\cos(\varphi), -d\sin(\varphi))$$

Onde:
- P é o ponto que desenha a curva;
- C é o centro do círculo interior;
- R e r são, respectivamente, os raios do círculo exterior e interior;
- d é a distância entre C e P;
- θ é o ângulo percorrido por C ao redor da origem;
- φ é o ângulo percorrido por P ao redor de C.

Desenhamos um exemplar dessa curva a seguir:

```
@variables θ R r d

C = (x=(R - r)*cos(θ), y=(R - r)*sin(θ));
ϕ = (R/r - 1)*θ;
P = (x=C.x + d*cos(ϕ), y=C.y - d*sin(ϕ));

(x, y) = substitute.((P.x, P.y), (Dict(
    R => 4,
    r => 1,
    d => 1
),))

points = curvepoints(x, y, θ, range(0, 2π, length=100));
vectors = tangvectors(x, y, θ, range(0, 2π, length=100));

(plt, anim) = genvis(points, vectors; xlim=(-5,5), ylim=(-5,5));

savefig(plt, "astroid.png");
gif(anim, "astroid.gif");

(plt, anim) = genvis(vectors; xlim=(-7,7), ylim=(-7,7));

gif(anim, "astvecs.gif"; fps=60);

points = curvepoints(x, y, θ, range(0, 15π, length=200));
vectors = tangvectors(x, y, θ, range(0, 15π, length=200));

(plt, anim) = genvis(points, vectors; xlim=(-5,5), ylim=(-5,5));
```

```
savefig(plt, "hypotch.png");
```

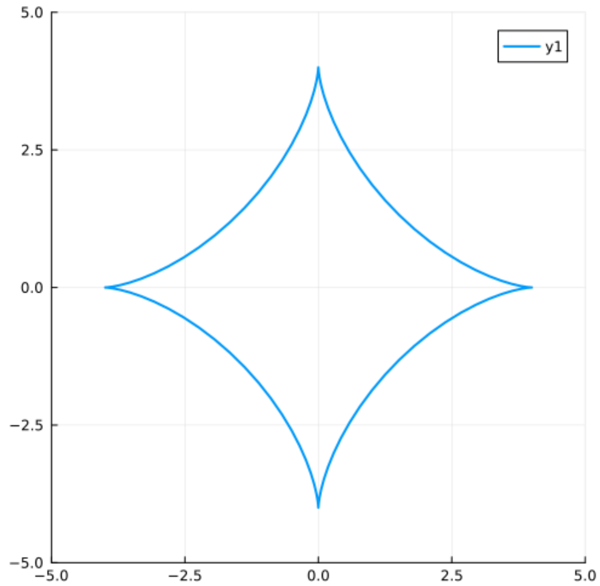

Figure 8: Astroide: Caso especial da hipotrocoide com $\frac{R}{r} = 4$ e $d = r$
Curva e vetor animados | Vetores acumulados

Questão 7

Para verificar que a curva $\alpha(t) = (a\cos(t), b\sin(t))$ é diferenciavel, verificamos que seus componentes são diferenciaveis:

$$\alpha_1'(t) = -a\sin(t)$$
$$\alpha_2'(t) = b\cos(t)$$

O traço é desenhado a seguir para $a = 2$ e $b = 3$:

```
@variables t a b

(x, y) = (a*cos(t), b*sin(t));
(x, y) = substitute.((x, y), (Dict(
    a => 2,
    b => 3
),));

points = curvepoints(x, y, t, range(0, 2π, length=100));

(plt, anim) = genvis(points; xlim=(-4,4), ylim=(-4,4));

savefig(plt, "elips.png");
```

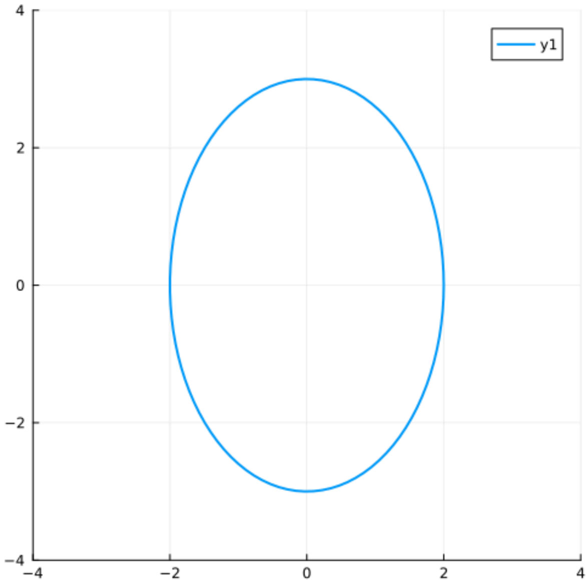

Figure 9: Elipse criada pela equação.

Questão 8

Começamos satisfazendo a condição $\alpha'(t) = (t^2, e^t)$. Para isso, tomamos a curva $\alpha(t) = \left(\frac{t^3}{3} + c_1, e^t + c_2\right)$. Em seguida, tomamos $c_1 = 2$ e $c_2 = -1$, satisfazendo a condição $\alpha(0) = (2, 0)$.

Questão 9

Começamos provando que $\|\alpha'(t)\|$ const. $\Rightarrow \langle \alpha''(t), \alpha'(t) \rangle = 0$. Tome a função $\varphi : I \to [0, 2\pi)$ que mapeia valores do parâmetro nos ângulos do vetor $\alpha'(t)$. Definindo $c = \|\alpha'(t)\|$, temos em coordenadas polares o seguinte:

$$\begin{aligned}\alpha'(t) &= (c\cos(\varphi(t)), c\sin(\varphi(t)))\\ \alpha''(t) &= (-c\sin(\varphi(t))\varphi'(t), c\cos(\varphi(t))\varphi'(t))\end{aligned}$$

Concluimos trivialmente que $\langle \alpha''(t), \alpha'(t) \rangle = 0$.

Em seguida, provaremos a volta $\langle \alpha''(t), \alpha'(t) \rangle = 0 \Rightarrow \|\alpha'(t)\|$ const.. Sejam $r = r(t)$ e $\theta = \theta(t)$ as coordenadas polares da curva. Logo:

$$\begin{aligned}\alpha' &= (r\cos(\theta), r\sin(\theta))\\ \alpha'' &= (r'\cos(\theta) - r\sin(\theta)\theta', r'\sin(\theta) + r\cos(\theta)\theta')\end{aligned}$$

Portanto, dado que α' e α'' são ortogonais:

$$\begin{gathered} rr'\cos(\theta)^2 - r^2\sin(\theta)\cos(\theta)\theta' + rr'\sin(\theta)^2 + r^2\sin(\theta)\cos(\theta)\theta' = 0\\ \Rightarrow rr'\left(\cos^2\theta + \sin^2\theta\right) = 0\\ \Rightarrow r = 0 \vee r' = 0\end{gathered}$$

Se $r = 0$, então nossa curva é apenas um ponto, um absurdo. Logo, concluimos que $r' = 0$ e, portanto, $\|\alpha'(t)\|$ const..

Questão 10

Começamos mostrando que x deve ser estritamente crescente ou decrescente. Tome x' e suponha que existam t_1 e t_2 tais que $x'(t_1)$ e $x'(t_2)$ tenham sinais opostos. Logo, pelo teorema do valor intermediário, sabemos que deve haver um ponto onde x' se anula. Isso é um absurdo pelo enunciado. Logo, x é estritamente crescente ou decrescente.

Podemos então afimar que existe uma bijeção entre x e t. Definimos então $t = t(x)$ e $f(x) = y(t(x))$. Como y é diferenciavel em relação a t e t em relação a x, concluimos que f é diferenciavel: $f'(x) = y'(t(x))t'(x)$.

Questão 11

Desenhamos essa curva γ da seguinte forma:

```
@variables t

γ = (x=e^t * cos(t), y=e^t * sin(t));

points = curvepoints(γ.x, γ.y, t, range(0, 8π, length=200));

function symgenvis(points)
   plt = plot(1; linewidth=2, size=(500,500), aspect_ratio=:equal, lims=:symmetric)
    anim = @animate for i = 1:size(points, 1)
        push!(plt, points[i,1], points[i,2])
    end

    return (plt, anim)
end

(plt, anim) = symgenvis(points);

savefig(plt, "spirl.png");
gif(anim, "spirl.gif"; fps=24);
```

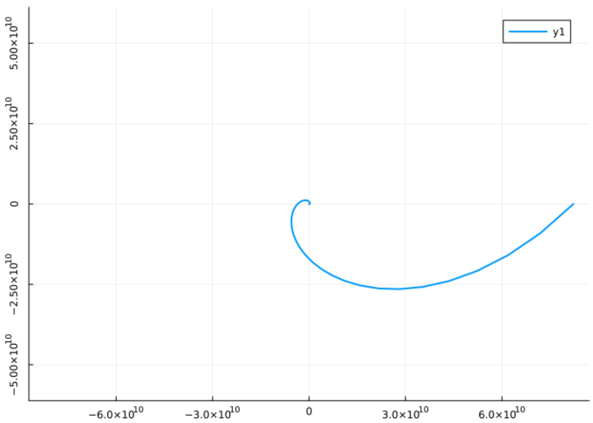

Figure 10: Espiral. Curva animada

Para mostrar a independência do ângulo em relação a t, calculamos as derivadas da curva γ:

$$x' = e^t \cos(t) - e^t \sin(t)$$
$$y' = e^t \cos(t) + e^t \sin(t)$$

Concluimos então que o produto interno entre γ e γ' é:

$$\begin{aligned}&\langle \gamma, \gamma' \rangle \\ &= e^{2t}(\cos^2 t - \cos t \sin t) \\ &+e^{2t}(\cos^2 t + \cos t \sin t) \\ &= e^{2t}\end{aligned}$$

$$\begin{aligned}&\langle \gamma, \gamma' \rangle \\ &= \|\gamma\| \|\gamma'\| \cos(\theta) \\ &= e^t \sqrt{2} e^t \cos(\theta) = e^{2t}\sqrt{2}\cos(\theta)\end{aligned}$$

$$\begin{aligned}&\Rightarrow e^{2t} = e^{2t}\sqrt{2}\cos(\theta) \\ &\Rightarrow \frac{1}{\sqrt{2}} = \cos(\theta) \\ &\Rightarrow \theta = \frac{\pi}{4}\end{aligned}$$

Concluimos, então, que o ângulo entre γ e γ' é fixo e independente de t.