3.4 Movimento ao longo de uma curva no espaço

O conceito de vetores e a notação vetorial, apresentados no capítulo 2.2, simplificam muito significativamente o estudo do movimento não-retilíneo, ou seja, o movimento em um plano ou, no caso mais geral, no espaço tridimensional.

Para descrever tal movimento ao longo de uma curva no plano ou no espaço tridimensional, precisamos falar sobre algumas propriedades adicionais das curvas, como a **curvatura** de uma curva em certo ponto da trajetória, o **raio de curvatura** ou o **círculo osculador**. Já ao discutir as ciclóides, na seçaõ anterior, mencionamos estes noções da geometria diferencial. Neste capítulo, vou discutir estes termos mais a fundo.
Em 3.4.1 vou representar os conceitos básicos que em **Com lápis e papel** serão detalhadamente explicados.

Os termos introduzidos serão ilustrados por meio de **Programas** de MuPAD.

3.4.1 Conceitos básicos

O comprimento do vetor-velocidade é v = ds/dt , o vetor **v** aponta no sentido do movimento.
Temos v = (**v·v**)$^{1/2}$, onde **v·v** é o produto escalar dos vetores **v**, **v**. O vetor

$$\bar{t} = \frac{\bar{v}}{v} = \frac{\bar{v}}{\dot{s}} = \frac{\bar{v}}{\frac{ds}{dt}} \qquad (3.4.1\text{-}1)$$

é unitário e tangente, tendo direção e sentido do movimento. No plano xy, **t** pode ser escrito como **t** = cos φ **i** + sen φ **j**, onde **φ** é o ângulo formado pelo eixo positivo X e **v**.

Muitas vezes se utiliza o comprimento de arco, s=s(t), como parâmetro em vez do parâmetro t (tempo). O vetor posição é, então, **x**(s) e a derivada desse vetor com referência a s denomina-se d**x**(s)/ds := **x**'(s). Em Física, prefere-se usar **r**(t) em vez de **x**(t).
Uma derivada com respeito do parâmetro t é, geralmente, denominada por um ponto acima da variável: $\dot{z}(t) := dz/dt$. Porém, é possível escrever z'(t), pois a indicação da variável diz claramente que se trata da derivada dz(t)/dt.

Já que s = s(t) é uma função composta, podemos estabelecer a seguinte relação entre **r**'(s) e **r**'(t), ou seja entre d**r**(s)/ds e d**r**(t)/dt:

$$\mathbf{r}'(s) = \mathbf{v}(t)/v \qquad (3.4.1\text{-}2)$$

Pois: $\mathbf{v}(t)=d\mathbf{r}(t)/dt = d\mathbf{r}(s)/ds \cdot ds/dt = d\mathbf{r}(s)/ds \cdot \dot{s} = \mathbf{r}'(s)\cdot v$

Se descrevermos a curva por meio do parâmetro s, teremos a seguinte expressão para o vetor unitário **t**:

$$\mathbf{t}(s) = \mathbf{r}'(s)/|\mathbf{r}'(s)| = \mathbf{r}'(s) \qquad (3.4.1\text{-}3)$$

já que $|\mathbf{r}'(s)| = 1$, veja "Com lápis e papel".

A derivada de **t** com respeito de s chama-se de **curvatura** κ :

$$\kappa = 1/\rho = |d\mathbf{t}/ds| = |\mathbf{t}'(s)| = |\mathbf{r}''(s)| \qquad (3.4.1\text{-}4)$$

Kápa (**κ**) é a curvatura e ro (**ρ**) o raio de curvatura do caminho percorrido.
Para o vetor unitário normal **n** temos a seguinte fórmula importante

$$\mathbf{n} = \rho\, \mathbf{r}''(s) \qquad (3.4.1\text{-}5)$$

Partindo de (3.4.1-2), obtemos para o vetor de curvatura **r**''(s)

$\mathbf{r}'' = d\mathbf{r}'/ds = d(\mathbf{v}(t)/v)/ds = d\mathbf{t}(t)/ds = d\mathbf{t}/dt \cdot 1/(ds/dt) = d\mathbf{t}/dt \cdot 1/(\mathbf{v}\cdot\mathbf{v})^{1/2}$, ou seja:

$$\mathbf{r}''(s) = d\mathbf{t}(t)/dt \cdot 1/(\mathbf{v}\cdot\mathbf{v})^{1/2} \qquad (3.4.1\text{-}6)$$

onde $(\mathbf{v}\cdot\mathbf{v})^{1/2}$ é o módulo $|\mathbf{v}|=v$ do vetor **v**.

Podemos usar também o produto vetorial (`linalg::crossProduct`) para o cálculo do valor absoluto de kápa e aplicar a seguinte expressão:

$$\kappa = \frac{\left|\bar{v} \times \bar{a}\right|}{\left(\sqrt{\bar{v} \cdot \bar{v}}\right)^3} \qquad (3.4.1\text{-}7)$$

onde o vetor **a**(t) é a aceleração.

Aínda temos outra equação para o vetor de curvatura:

$$\boldsymbol{\kappa} = (\mathbf{a} - \mathbf{v} \cdot dv/dt \cdot 1/v)/v^2 \qquad (3.4.1\text{-}8)$$

e temos uma última relação para o raio de uma curva {x = x(t), y = y(t)} no plano-xy:

$$\rho = (x'^2 + y'^2)^{3/2} / (x'y'' - y'x'') \qquad (3.4.1\text{-}9)$$

No caso de uma elipse {x = a cos t, y = b sen t}, temos

$$\rho = (a^2 \text{ sen}^2\, t + b^2 \cos^2 t)^{3/2} / (a\, b) \qquad (3.4.1\text{-}10)$$

Todas estas fórmulas nos oferecem não só uma maneira de calcular kápa e ro, mas também são excelentes exemplos para praticar o uso de MuPAD.

3.4.2 Programas

No **primeiro** programa calculamos a curvatura e o raio de curvatura de uma elipse, utilizando quatro diferentes fórmulas: (3.4.1-6), (3.4.1-7), (3.4.1-8), (3.4.1-10).

O **segundo** programa fabrica uma representação gráfica do vetor-curvatura no caso de uma elipse. Confira bem o comprimento de **r**" com o grau de curvatura da trajetória.

Com o **terceiro** programa fazemos uma animação que demonstra os vetores **v** e **r"** ao longo de uma elipse. Observe o comprimento (módulo) dos vetores nos diferentes pontos da trajetória. Onde a curvatura é grande, a velocidade é pequena e vice-versa!

O **quarto** programa mostra o **círculo osculador** num ponto de uma elipse. Aplique as programas usando diferentes valores iniciais.

Programa 1:

```
• reset()://curvatura com varias fórmulas
t1:=PI/7:
a:=9:b:=5:c:=0:// elipse, já que c=0
x:=t->a*cos(t):
y:=t->b*sin(t):
z:=t->c*t:// fórmulas para uma hélice
pos:=matrix([[x(t),y(t),z(t)]])://vetor-posição
v:=matrix([[x'(t),y'(t),z'(t)]])://vetor-velocidade
ac:=matrix([[x''(t),y''(t),z''(t)]])://vetor-aceleração
absolv:=sqrt(v[1,1]^2+v[1,2]^2+v[1,3]^2):
derv:=diff(absolv,t)://d|v|/dt
vt:=v/absolv://vetor-tangente-unitário
vc:=diff(vt,t)/absolv://vetor-curvatura (3.4.1-6)
cross:=linalg::crossProduct(v,ac):
abscross:=sqrt(linalg::scalarProduct(cross,cross)):
t:=t1:
absolvc:=sqrt(vc[1,1]^2+vc[1,2]^2+vc[1,3]^2):
kk:=ac/absolv^2-derv*v/absolv^3://vetor-curvatura-(3.4.1-8)
absolk:=float(abscross/absolv^3)://(3.4.1-7)
//raios de curvatura:
ro:=1/absolk;
ro1:=1/float(absolvc);//raio de curvatura
ro2:=1/float(sqrt(kk[1,1]^2+kk[1,2]^2+kk[1,3]^2));
ro2d:=float((a^2*sin(t)^2+b^2*cos(t)^2)^1.5/(a*b));
//Eq.(3.4.1-10)
```

Resultados :

4.708748705

4.708748705

4.708748705

4.708748705

Programa 2:

```
• reset()://gráfico do vetor-curvatura
  t1:=PI/5:a:=9: b:=5:
  x:=t->a*cos(t):
  y:=t->b*sin(t):
  scale:=20://elipse
  curve:=plot::Curve2d([x(t),y(t)],t=0..PI)://pano de fundo
  pos:=matrix([[x(t),y(t)]])://vetor-posição
  v:=matrix([[x'(t),y'(t)]])://vetor-velocidade
  absolv:=sqrt(v[1,1]^2+v[1,2]^2):
  vt:=v/absolv://vetor-tangente-unitário
  k:=diff(vt,t)/absolv://vetor de curvatura, eq. (14)
  t:=t1:
  x1:=pos[1,1]:y1:=pos[1,2]:
  x2:=x1+ scale*k[1,1]:
  y2:=y1+ scale*k[1,2]:
```

```
p:=plot::Point2d(x1,y1,Color=RGB::Green,PointSize=2*unit::m
m):

kc:=plot::Arrow2d([x1,y1],[x2,y2],Color=RGB::Red):

plot(curve,p,kc,Scaling=Constrained)
```

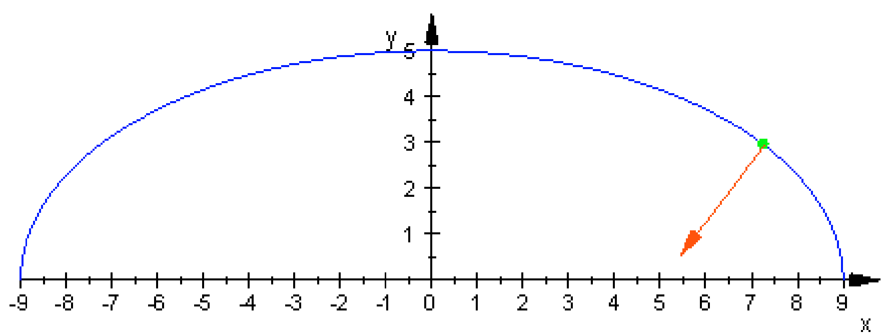

Fig. 3.4-1

Program 3 :

- ```
  reset()://curvatura e velocidade com animação
    //t1:=PI/2:
  a:=9: b:=5:
  x:=t->a*cos(t):
  y:=t->b*sin(t):
  scale:=20://elipse
  curve:=plot::Curve2d([x(t),y(t)],t=0..2*PI):
  pos:=matrix([[x(t),y(t)]])://vetor-posição
  v:=matrix([[x'(t),y'(t)]])://vetor-velocidade
  ```
  ```

```
absolv:=sqrt(v[1,1]^2+v[1,2]^2)://módulo de v

vt:=v/absolv://vetor-tangente-unitário

k:=diff(vt,t)/absolv://vetor de curvatura

x1:=pos[1,1]:

y1:=pos[1,2]:

v1:=v[1,1]:

v2:=v[1,2]:

x2:=x1+0.5*v1://extremo do vetor v com fator de redução

y2:=y1+0.5*v2:

x3:=x1+scale*k[1,1]://extremo do vetor k

y3:=y1+ scale*k[1,2]:

p:=plot::Point2d(x1,y1,t=0..2*PI,Color=RGB::Green,PointSize
=2*unit::mm):

vc:=plot::Arrow2d([x1,y1],[x2,y2],t=0..2*PI,Color=RGB::Blac
k)://velocidade

kc:=plot::Arrow2d([x1,y1],[x3,y3],t=0..2*PI,Color=RGB::Red)
://curvatura

plot(curve,p,vc,kc,Scaling=Constrained)
```

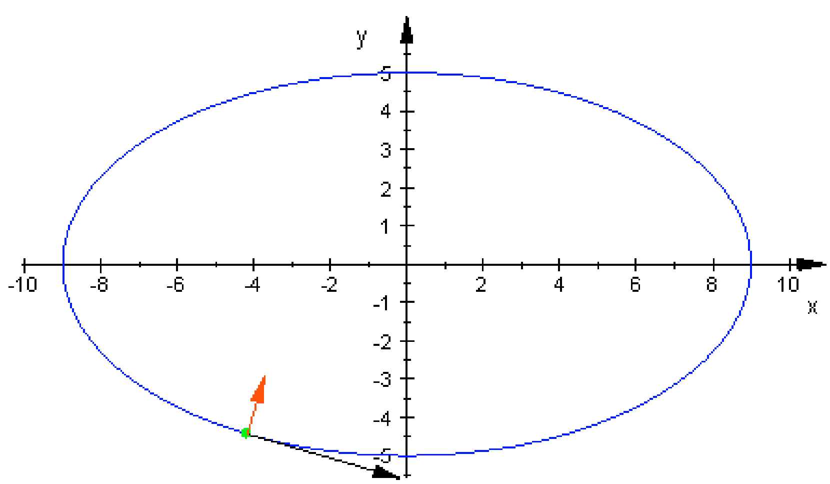

Fig. 3.4-2

Programa 4:

```
reset()://curvatura com círculo-osculador
t1:=5*PI/6: a:=3: b:=2:
x:=t->a*cos(t):
y:=t->b*sin(t):
scale:=4://elipse
curve:=plot::Curve2d([x(t),y(t)],t=0..PI):
pos:=matrix([[x(t),y(t)]])://vetor-posição
v:=matrix([[x'(t),y'(t)]])://vetor-velocidade
absolv:=sqrt(v[1,1]^2+v[1,2]^2):
et:=v/absolv://vetor-tangente-unitário
k:=diff(et,t)/absolv://vetor de curvatura
raio:=1/sqrt(k[1,1]^2+k[1,2]^2):
centro:=pos+k*raio^2:
t:=t1:
x1:=pos[1,1]:
y1:=pos[1,2]:
x2:=x1+scale*k[1,1]:
y2:=y1+scale*k[1,2]:
x3:=centro[1,1]:
y3:=centro[1,2]:
circulo:=plot::Circle2d(raio,[x3,y3],Color=RGB::Black):
pc:=plot::Point2d(x3,y3,Color=RGB::Black,PointSize=2*unit::
mm):
p:=plot::Point2d(x1,y1,Color=RGB::Green,PointSize=2*unit::m
m):
```

```
kc:=plot::Arrow2d([x1,y1],[x2,y2],Color=RGB::Red):
plot(curve,pc,p,kc,circulo,Scaling=Constrained)
```

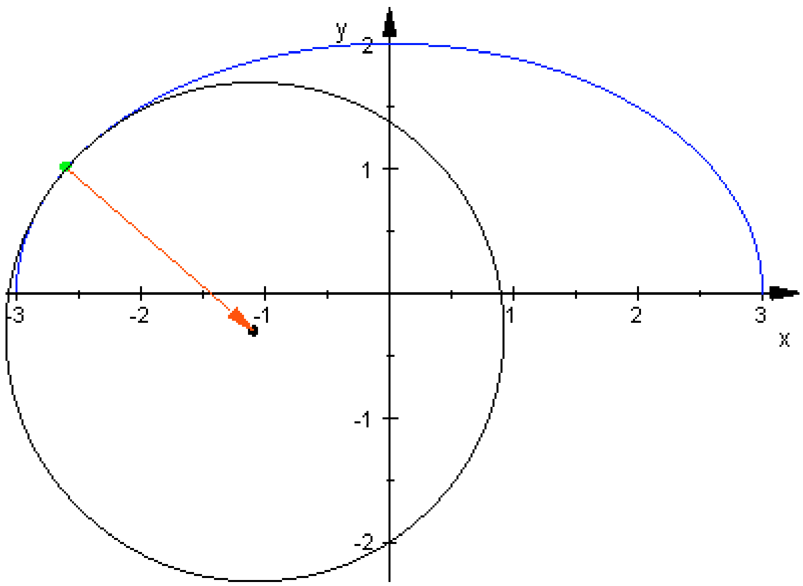

Fig. 3.4-3

3.4.3 Com lápis e papel

Movimento curvilíneo: Velocidade

Vamos agora considerar uma partícula descrevendo uma trajetória curvilínea C. No instante t, a partícula está no Ponto A, dado pelo vetor-posição **r**(t) = x(t)**i** + y(t)**j** +z(t)**k**.

Quando t mudar, o ponto extremo do vetor **r** vai descrever uma curva no espaço que tem a representação {x=x(t), y = y(t), z = z(t)}.

Embora o movimento da partícula seja ao longo do arco AB = Δs, o deslocamento designamos, porém, pelo vetor **AB** = Δ**r** = **r**(t + Δt) - **r**(t).

A velocidade média Δ**r**/Δt = (**r**(t + Δt)-**r**(t))/Δt ı um vetor na direção de Δ**r**.

Fazendo Δt muito pequeno, vamos obter a velocidade instantβnea

$$\lim_{\Delta t \to 0} \frac{\Delta \bar{r}}{\Delta t} = \frac{d\bar{r}}{dt} = \bar{v} \qquad (1)$$

No limite, quando B está bem próximo de A, o vetor Δ**r** coincide em direção com a tangente no ponto A. Assim, no movimento curvilíneo, a velocidade instantânea é um vetor tangente à trajetória.

Há outra maneira para localizar uma partícula sobre uma trajetória, a saber, por meio do deslocamento s = O*A ao longo da curva. O* é um ponto de referência arbitrário sobre a trajetória.

Quando a partícula vai de A a B, o deslocamento Δs ao longo da curva ı dado pelo comprimento do arco AB.

Multiplicando e dividindo a equação (1) por Δs, obtemos

$$\bar{v} = \lim_{\Delta t \to 0} \left(\frac{\Delta \bar{r}}{\Delta s} \cdot \frac{\Delta s}{\Delta t}\right) = \lim_{\Delta s \to 0} \frac{\Delta \bar{r}}{\Delta s} \cdot \lim_{\Delta t \to 0} \frac{\Delta s}{\Delta t} \qquad (2)$$

visto que Δs → 0 quando Δt → 0.
O módulo de Δ**r** é, aproximadamente, igual a Δs, e quanto mais prσximo de A estiver B, mais próximo do valor de Δs estará o módulo de Δ**r**,
Portanto, o fator

$$\lim_{\Delta s \to 0} \frac{\Delta \bar{r}}{\Delta s} = \frac{d\bar{r}}{ds} = \bar{r}\,' = \hat{t}(s) \qquad (3)$$

representa um vetor de módulo unitário e direção tangente à trajetória. Assim, a equação (2) pode-se escrever como

v = ds/dt · **t** = v·**t**, onde **t** é o vetor tangente unitário: **t** = **v**(t)/v(t).

A taxa de variação de **t** com relação a s é uma medida da curvatura da trajetória num ponto **r** e é dada pelo vetor de curvatura **κ**. Utilizando a equação (3), obtemos

κ := d**t**/ds = **r**'' (4)

A direção de **κ,** em cada ponto da curva, é perpendicular (normal) à curva no ponto dado.

Se **n** for um vetor unitário nesta direção normal, podemos escrever

$$d\mathbf{t}/ds = \kappa \cdot \mathbf{n} \qquad (5)$$

O número kápa é a *curvatura*, e **n** é chamado de vetor unitário normal principal no sentido da concavidade da curva.

A grandeza **ρ := 1/κ** tem o nome de *raio de curvatura*.

Um vetor unitário **b**, normal ao plano definido por **t** e **n** com **b** = **t** x **n** é chamado de *binormal* à curva.

(**t**, **n**, **b**) é uma tripla positiva de vetores unitários, eles formam um sistema de coordenadas ortogonais em cada ponto especificado da trajetória.

Há três equações fundamentais que contêm as derivadas de (**t**, **n**, **b**) com relação a s:

$$d\mathbf{t}/ds = \kappa\, \mathbf{n} \qquad (6)$$

$$d\mathbf{n}/ds = \tau\, \mathbf{b} - \kappa\, \mathbf{t} \qquad (7)$$

$$d\mathbf{b}/ds = - \tau\, \mathbf{n} \qquad (8)$$

τ (tau) tem o nome de *torsão*. O plano definido pelos vetores **t, n** chama-se de *plano osculador* da curva em A.

Movimento curvilíneo: Aceleração

Quando a velocidade de uma partícula varia de $\mathbf{v}_1$ para $\mathbf{v}_2$ em um intervalo de tempo Δt, sua aceleraηão média durante Δt é $\mathbf{a}_{med}$ = ($\mathbf{v}_2$ -$\mathbf{v}_1$)/Δt = Δ**v**/Δt e a aceleraηão instantânea será

$$\lim_{\Delta t \to 0} \frac{\Delta \bar{v}}{\Delta t} = \bar{a} = \frac{d\bar{v}}{dt} = \frac{d^2\bar{r}}{dt^2} \qquad (9)$$

A velocidade pode variar em módulo, direção ou sentido (ou em mais de um). A aceleração é um vetor que tem a direção da variação instantânea da velocidade e é sempre dirigida para a concavidade da curva.

Visto que **a** está dirigida para a concavidade da trajetória, podemos decompô-la numa componente tangencial $\mathbf{a}_t$ e uma componente normal $\mathbf{a}_n$. A componente tangencial é a causa para a variação do módulo da velocidade. A componente normal é a causa para a variação da direção da velocidade.

Temos **a** = d**v**/dt = d(v·**t**)/dt = dv/dt· **t** + v·d**t**/dt. Se a trajetória fosse uma reta, o vetor **t** seria constante em módulo e direção, e d**t**/dt seria zero.

Quando a trajetória é curva, a direção de **t** varia ao longo da curva e d**t**/dt não será nulo.

Para calcular a derivada de **t**, escrevemos (sendo ρ = 1)

$$\mathbf{t} = \cos\varphi\ \mathbf{i} + \text{sen}\ \varphi\ \mathbf{j}$$

$$\mathbf{n} = \cos(\varphi+\pi/2)\ \mathbf{i} + \text{sen}\ (\varphi+\pi/2)\ \mathbf{j}$$

$$= -\ \text{sen}\ \varphi\ \mathbf{i} + \cos\varphi\ \mathbf{j}$$

sendo φ o βngulo que a tangente à curva em A faz com o eixo X.

(A componente de um vetor, numa dada direção, é igual à projeção do vetor naquela direção.)

Então, d**t**/dt = -sen φ · dφ/dt **i** + cos φ · dφ/dt · **j**, ou seja

$$\frac{d\hat{t}}{dt} = \frac{d\varphi}{dt} \cdot \hat{n} \qquad (10)$$

é normal à curva.

Nós nos lembramos de que φ:= s/r em radianos, ou dφ = ds/ρ, onde ρ ı o raio de curvatura. Por isso temos dφ/dt = dφ/ds · ds/dt = v/ρ e a equaηão (10) se torna em

$$\frac{d\hat{t}}{dt} = \frac{v}{\rho} \cdot \hat{n} \qquad (11)$$

Assim obtemos, finalmente,

$$\bar{a} = \frac{dv}{dt} \cdot \hat{t} + \frac{v^2}{\rho} \cdot \hat{n} \qquad (12)$$

O primeiro termo é a aceleração tangencial com a_t := dv/dt = d^2s/dt^2 , o segundo termo é a aceleração normal com a_n := $v^2/\rho = (ds/dt)^2/\rho$.

$\mathbf{a}_t$ e $\mathbf{a}_n$ são sempre perpendiculares entre si.

No caso de um **movimento circular** uniforme, temos s = r·φ e dφ/dt ı a velocidade angular ω da partνcula.

Para a aceleração temos: $\mathbf{a}(t) = \omega \cdot v \cdot \mathbf{n}(t) = v^2/r \cdot \mathbf{n} = \omega^2 \cdot r \cdot \mathbf{n} = - \omega^2 \cdot \mathbf{r}(t)$.

Vê-se que a aceleração **a**(t) aponta para o centro do círculo, fala-se, por isso, de aceleração *centrípeta* ("que busca o centro"). Se a trajetória fosse uma curva arbitrária, o vetor **n** apontaria para o centro do círculo osculador, confira com a figúra (3.4.-3).

Curvatura de uma trajetória

Voltemos, mais uma vez, à equação (5): $d\mathbf{t}/ds = \mathbf{r}'' = \kappa \cdot \mathbf{n}$ com $\mathbf{t} = d\mathbf{r}/ds := \mathbf{r}'$

Derivando a relativo ao tempo, obteremos

$d\mathbf{t}/dt = d\mathbf{r}'/dt = d\mathbf{r}'/ds \cdot ds/dt = \mathbf{r}'' \cdot v$, ou seja

$$\frac{d\hat{t}}{dt} = \bar{r}'' \cdot v \qquad (13)$$

Desta equação obtemos para o vetor-curvatura

$$\bar{r}'' = \frac{1}{v} \cdot \frac{d\hat{t}}{dt} \qquad (14)$$

No primeiro e segundo dos programas determinamos por meio desta fórmula o vetor curvatura.
Para calcular **r**", precisamos de determinar $v = (v_x^2 + v_y^2 + v_z^2)^{1/2}$ e a derivada do vetor **t** = **v**(t)/v(t) com respeito do tempo. As instruções nestes programas são :

```
absolv:=sqrt(v[1,1]^2+v[1,2]^2+v[1,3]^2)://módulo de v

derv:=diff(absolv,t)://d|v|/dt

vt:=v/absolv://vetor-tangente-unitário t

vc:=diff(vt,t)/absolv://vetor-curvatura (3.4.1-6)
```

O módulo |**v**| podemos, em princípio, também calcular por meio do produto escalar **v** · **v**, mas, parece, que MuPAD não gosta, neste caso, desse método.

O vetor **r**" pode-se calcular também em forma direta, obtendo o resultado

$$\bar{r}'' = \frac{v \cdot \dot{\bar{v}} - \dot{v} \cdot \bar{v}}{v^2} \cdot \frac{1}{v} = \frac{1}{v^2} \cdot \bar{a} - \frac{\dot{v} \cdot \bar{v}}{v^3} \qquad (15)$$

Você encontra esta expressão no primeiro programa.
Para mostrar a equação (15), só tem de fazer o seguinte cálculo:

$\mathbf{r}'' = d\mathbf{r}'/ds = d(\mathbf{t}(s))/ds = d(\mathbf{v}(t)/v(t))/ds = d(\mathbf{v}(t)/v(t))/dt \cdot 1/(ds/dt)$

A expressão (14), $\mathbf{r}'' = 1/v \cdot d\mathbf{t}/dt$, não é sempre a mais conveniente para o cálculo do valor da curvatura. Na prática é mais fácil usar a seguinte fórmula

$$\kappa(\mathbf{t}) = \frac{|\bar{a}(\mathbf{t}) \times \bar{v}(\mathbf{t})|}{v^3(\mathbf{t})} \qquad (16)$$

Também esta expressão foi aplicada no primeiro programa. A sua demonstração não é muito difícil: temos de multiplicar a eq. (12) com $\mathbf{t} = \mathbf{v}(t)/v(t)$

$$\begin{aligned} \mathbf{a} \times \mathbf{v} &= (v'(t)\cdot \mathbf{t}(t) + \kappa(t) \cdot v^2(t)\cdot\mathbf{n}(t)) \times v(t)\cdot\mathbf{t}(t) \\ &= v'\cdot v\cdot\mathbf{t} \times \mathbf{t} + \kappa\cdot v^3\cdot\mathbf{n} \times \mathbf{t} \\ &= \kappa\cdot v^3\cdot\mathbf{n} \times \mathbf{t} \end{aligned}$$

$|\mathbf{a} \times \mathbf{v}| = \kappa\cdot v^3\cdot|\mathbf{n}|\cdot|\mathbf{t}|\cdot \text{sen}\,(\pi/2) = \kappa\cdot v^3$

Finalmente determinamos o centro do círculo osculador **m** para o ponto A na trajetória.

Da seguinte figura vemos que $\mathbf{m} = \mathbf{r} + \rho\,\mathbf{n}$.

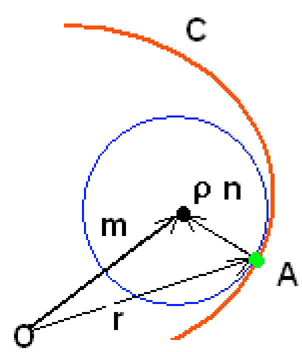

Fig. 3.4-4

Problema:

Fazer um gráfico com três vetores de curvatura, semelhante ao da Fig. 3.4-1

Proposta de uma **solução** (veja também a próxima seção):

```
• reset()://gráfico com três vetores
  t1:=PI/5:t2:=PI/3:t3:=PI/2:
  a:=9: b:=5:
  x:=t->a*cos(t):
  y:=t->b*sin(t):
  scale:=20://elipse
  curve:=plot::Curve2d([x(t),y(t)],t=0..PI):
  pos:=matrix([[x(t),y(t)]])://vetor-posição
  v:=matrix([[x'(t),y'(t)]])://vetor-velocidade
  absolv:=sqrt(v[1,1]^2+v[1,2]^2):
  vt:=v/absolv://vetor-tangente-unitário
  k:=diff(vt,t)/absolv://vetor de curvatura, eq. (14)
  t:=t1:
  x11:=pos[1,1]:y11:=pos[1,2]:
  x21:=x11+ scale*k[1,1]:y21:=y11+ scale*k[1,2]:
  t:=t2:
  x12:=pos[1,1]:y12:=pos[1,2]:
  x22:=x12+ scale*k[1,1]:y22:=y12+ scale*k[1,2]:
  t:=t3:
  x13:=pos[1,1]:y13:=pos[1,2]:
  x23:=x13+ scale*k[1,1]:y23:=y13+ scale*k[1,2]:
```

```
p1:=plot::Point2d(x11,y11,Color=RGB::Green,PointSize=2*unit
::mm):

p2:=plot::Point2d(x12,y12,Color=RGB::Green,PointSize=2*unit
::mm):

p3:=plot::Point2d(x13,y13,Color=RGB::Green,PointSize=2*unit
::mm):

kc1:=plot::Arrow2d([x11,y11],[x21,y21],Color=RGB::Red):

kc2:=plot::Arrow2d([x12,y12],[x22,y22],Color=RGB::Red):

kc3:=plot::Arrow2d([x13,y13],[x23,y23],Color=RGB::Red):

plot(curve,p1,p2,p3,kc1,kc2,kc3,Scaling=Constrained)
```

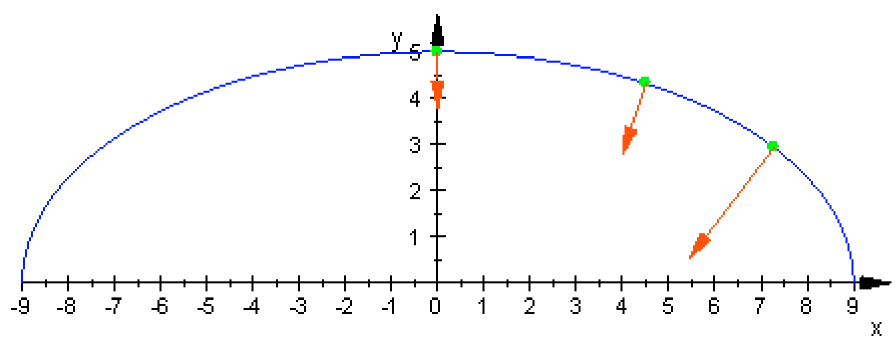

Fig. 3.4-5

Esta seção é muito grande. Vou dividi-la em duas partes. Na próxima parte vamos retomar os programas anteriores e introduzir algumas técnicas e matérias novas.