Kurvor, fält och ytor

1 Inledning

Vi skall se hur man ritar parametriserade kurvor i planet $\boldsymbol{r} : \mathbb{R} \to \mathbb{R}^2$ och i rummet $\boldsymbol{r} : \mathbb{R} \to \mathbb{R}^3$. Därefter skall vi approximera en kurvas båglängd.

Sedan skall vi rita några vektorfält och deras strömlinjer. Vidare ser vi hur vi kan approximera arean av ett område i planet. Avslutningsvis ritar vi några parametriserade ytor $\boldsymbol{r} : \mathbb{R}^2 \to \mathbb{R}^3$ i rummet.

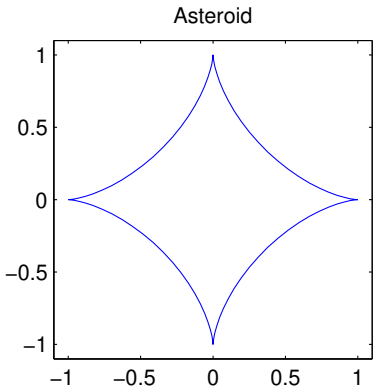
2 Kurvor i $\mathbb{R}^2$ och $\mathbb{R}^3$

Vi har tidigare ritat kurvor i $\mathbb{R}^2$ med kommandot `plot`. Som ett exempel tar vi asteroiden

$$\boldsymbol{r}(t) = (x(t), y(t)) = (\cos^3(t), \sin^3(t)), \quad 0 \leq t \leq 2\pi$$

som vi ritar upp i MATLAB enligt

```
>> t=linspace(0,2*pi);
>> x=cos(t).^3; y=sin(t).^3;
>> plot(x,y)
>> axis equal, axis([-1.1 1.1 -1.1 1.1])
>> title('Asteroid')
```



Lägg märke till att vi använder `axis equal` för att få rätt utseende (aspect ratio) och för att få lite "luft" runt kurvan använder vi `axis([...])`. Vidare upphöjer vi med komponentvisa operationer (`t` är ju en vektor).

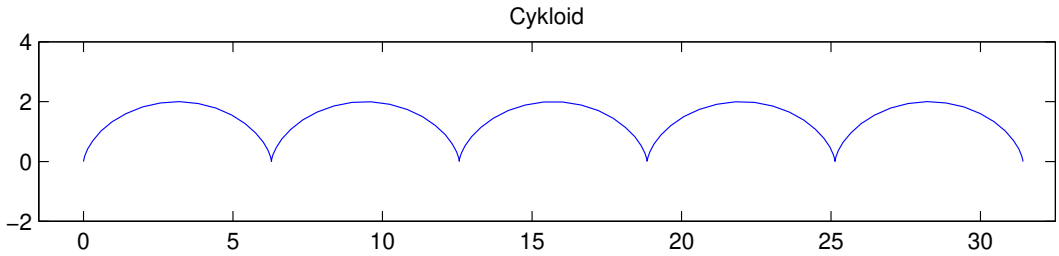
Ett annat exempel är cykloiden

$$\boldsymbol{r}(t) = (x(t), y(t)) = (t - \sin(t), 1 - \cos(t)), \quad 0 \leq t \leq 10\pi$$

som beskriver den väg en myra, som fastnat på ett hjul, färdas när hjulet rullar framåt.

Vi ritar enligt

```
>> t=linspace(0,10*pi);
>> plot(t-sin(t),1-cos(t))
>> axis equal, axis([-1.5 32.5 -2 4])
>> title('Cykloid')
```



Om du vill kan du köra skriptet `myran`, som du finner på laborationshemsidan, så du ser hjulet rulla.

Uppgift 1. Skriv upp en parametrisering av en ellips med storaxel a (x-riktningen) och lillaxel b (y-riktningen) och medelpunkt i (p, q). Låt $a = 1$, $b = 0.5$ och $p = q = 0$. Rita en bild av kurvan. Använd `axis equal`.

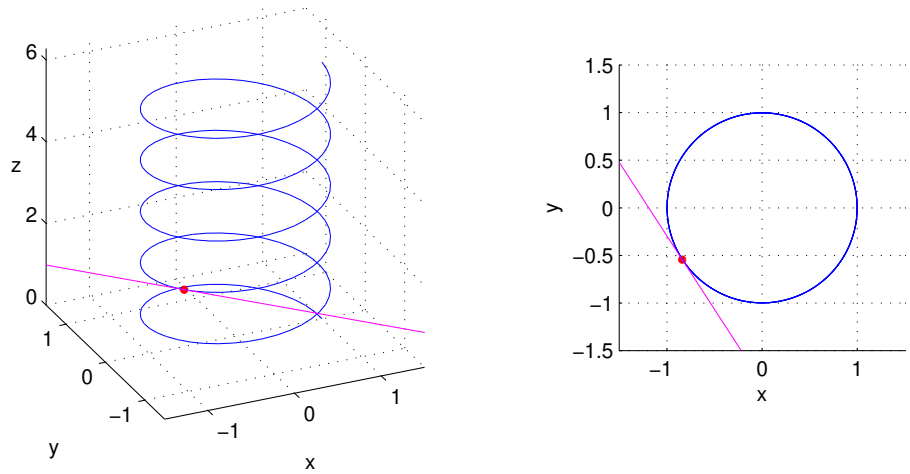
Som ett exempel på att rita kurvor i $\mathbb{R}^3$ tar vi spiralen

$$\boldsymbol{r}(t) = (x(t), y(t), z(t)) = (\cos(ct), \sin(ct), t), \quad 0 \leq t \leq 2\pi$$

där vi tar $c = 5$. Nu skall vi använda `plot3` för att rita kurvan

```
>> c=5; t=linspace(0,2*pi,200);
>> x=cos(c*t); y=sin(c*t); z=t;
>> plot3(x,y,z)
>> grid on
>> xlabel('x'), ylabel('y'), zlabel('z','rotation',0)
>> axis([-1.5 1.5 -1.5 1.5 0 2*pi])
>> axis vis3d, rotate3d on
```

Vi ser kurvan nedan till vänster. Där har vi även passat på att rita tangenten i en punkt.



Vi använder `axis vis3d` så att skalan inte förändras då vi vrider och vänder på grafen, med `rotate3d on` blir det möjligt att ta tag i grafen och vrida den.

För att rita tangenten får vi beräkna derivatan

$$\boldsymbol{r}'(t) = (x'(t), y'(t), z'(t)) = (-c\sin(ct), c\cos(ct), t)$$

för att sedan rita upp den räta linjen med

```
>> a=2; s=[-1 1];
>> xa=cos(c*a); ya=sin(c*a); za=a;
>> dxa=-c*sin(c*a); dya=c*cos(c*a); dza=1;
>> hold on
>> plot3(xa+s*dxa,ya+s*dya,za+s*dza,'m')
>> plot3(xa,ya,za,'ro','markersize',2,'linewidth',2)
>> hold off
```

Bilden ovan till höger ser vi kurvan rakt ovanifrån och ser då tydligt hur tangenten just tangerar kurvan som den skall.

Uppgift 2. Rita upp följande kurvor i $\mathbb{R}^3$ med `plot3`. Använd `axis vis3d` och `rotate3d on`.

(a). $\boldsymbol{r}(t) = (x(t), y(t), z(t)) = (\cos(10t), \sin(30t), t), \quad 0 \le t \le 2\pi$

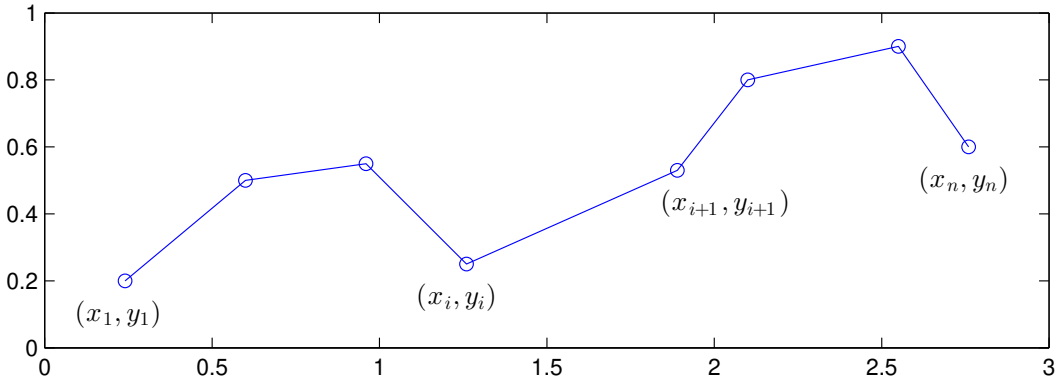
(b). $\boldsymbol{r}(t) = (x(t), y(t), z(t)) = (\cos^3(10t), \sin^3(10t), t), \quad 0 \le t \le 2\pi$

3 Båglängd och polygontåg

Vi tänker oss att vi har ett polygontåg

$$(x_1, y_1),\ (x_2, y_2),\ \cdots,\ (x_i, y_i),\ (x_{i+1}, y_{i+1}),\ \cdots,\ (x_n, y_n)$$

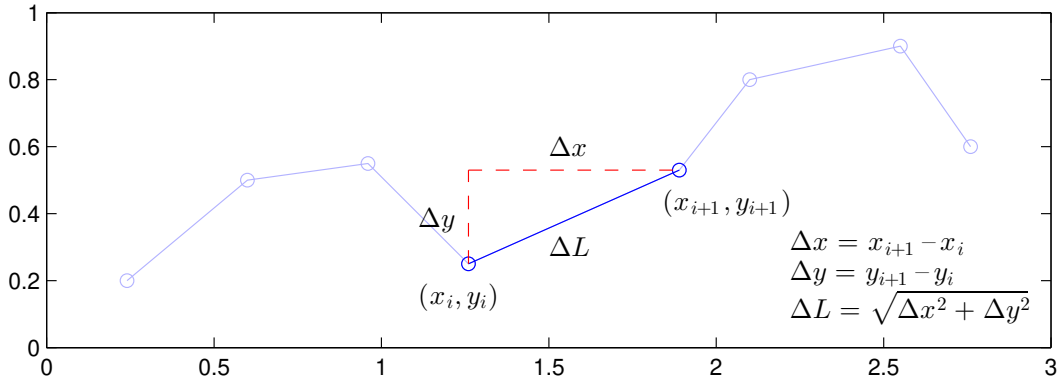
som vi ritat en figur av



Vill vi beräkna polygontågets längd kan vi göra det med formeln

$$L = \sum_{i=1}^{n-1} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$$

Denna formel, som vi såg på redan i "Kontrollstrukturer i MATLAB" i den inledande matematikkursen, fås genom att använda Pytagoras sats på varje segment i polygontåget.



Antag att koordinaterna samlade i två vektorer $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ och $\mathbf{y} = (y_1, y_2, \cdots, y_n)$, då beräknar vi längden enligt

```
>> n=length(x);
>> L=0;
>> for i=1:n-1
       L=L+sqrt((x(i+1)-x(i))^2+(y(i+1)-y(i))^2);
   end
>> L
```

eller lite kortare med vektorisering

```
>> L=sum(sqrt((x(2:end)-x(1:end-1)).^2+(y(2:end)-y(1:end-1)).^2))
```

alternativt

```
>> L=sum(sqrt(diff(x).^2+diff(y).^2))
```

Med beteckningen $\boldsymbol{r}_i = (x_i, y_i)$ har vi

$$L = \sum_{i=1}^{n-1} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} = \sum_{i=1}^{n-1} \parallel \boldsymbol{r}_{i+1} - \boldsymbol{r}_i \parallel$$

Om vi nu har en parametriserad kurva $\boldsymbol{r}(t) = (x(t), y(t))$, $a \leq t \leq b$ så kan vi beräkna en approximation av båglängden om vi tar en antal punkter längs kurvan och beräkna längden av motsvarande polygontåg.

Låt $a = t_1 < t_2 < \cdots < t_n = b$, med $\Delta t_i = t_{i+1} - t_i$, vara en indelning av parameterintervallet och låt

$$\boldsymbol{r}_i = \boldsymbol{r}(t_i) = (x(t_i), y(t_i))$$

och beräknar

$$s_n = \sum_{i=1}^{n-1} \parallel \boldsymbol{r}_{i+1} - \boldsymbol{r}_i \parallel$$

Detta är en approximation av båglängden. Ju fler punkter, desto noggrannare resultat.

Vi har

$$s_n = \sum_{i=1}^{n-1} \parallel \boldsymbol{r}_{i+1} - \boldsymbol{r}_i \parallel = \sum_{i=1}^{n-1} \left\| \frac{\boldsymbol{r}_{i+1} - \boldsymbol{r}_i}{\Delta t_i} \right\| \Delta t_i$$

så om $\boldsymbol{r}'(t)$ kontinuerlig får vi kurvans längd

$$L = s = \lim_{\substack{n\to\infty \\ \max \Delta t_i \to 0}} \sum_{i=1}^{n-1} \left\| \frac{\boldsymbol{r}_{i+1} - \boldsymbol{r}_i}{\Delta t_i} \right\| \Delta t_i = \int_a^b \| \boldsymbol{r}'(t) \| \, dt$$

som vi känner igen från Adams kapitel 11.3.

Vi får motsvarande resultat för polygontåg och kurvor i rummet.

Uppgift 3. Bestäm en approximation av omkretsen av ellipsen i uppgift 1. Det var ett polygontåg ni ritade upp. Börja med ett fåtal punkter (10 kanske) i polygontåget och öka successivt antalet (ända upp till 100 säg) och se hur approximationen av omkretsen förbättras.

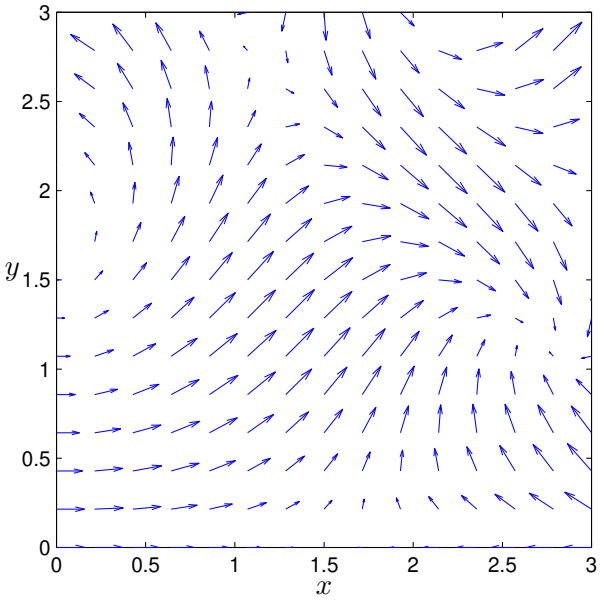
4 Vektorfält i planet

Vi ser på ett vektorfält i planet, dvs. en funktion $\boldsymbol{F}\colon \mathbb{R}^2 \to \mathbb{R}^2$. Som exempel tar vi

$$\boldsymbol{F}(x, y) = (F_1(x, y), F_2(x, y)) = (\cos(x - y), \sin(xy))$$

Vi ritar upp fältet med `quiver` enligt

```
>> [X,Y]=meshgrid(linspace(0,3,15),linspace(0,3,15));
>> quiver(X,Y,cos(X-Y),sin(X.*Y),0.8)
>> axis equal, axis([0 3 0 3])
```



Som ni säkert vet är fältlinjer (strömlinjer, flödeslinjer, kraftlinjer) till ett vektorfält $\boldsymbol{F}$ kurvor $\boldsymbol{r}(t) = (x(t), y(t))$ vars tangenter $\boldsymbol{r}'(t) = (x'(t), y'(t))$ är parallella med vektorfältets vektorer $\boldsymbol{F}(\boldsymbol{r}(t)) = \boldsymbol{F}(x(t), y(t))$, dvs. kurvor som följer fältet.

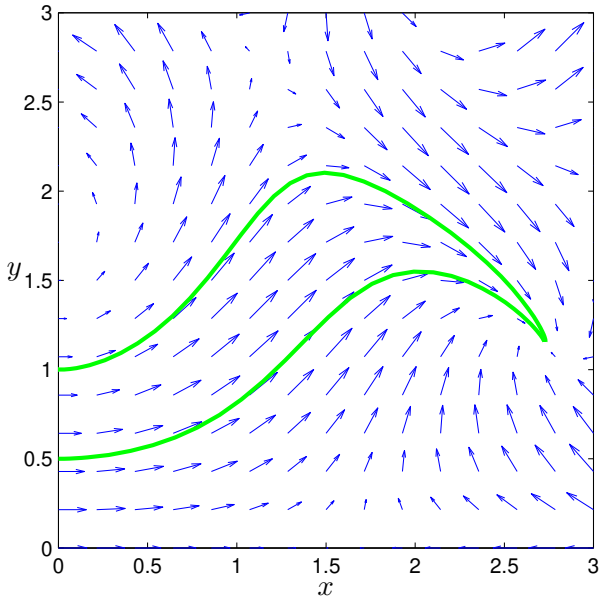
Vi kan beräkna fältlinjer genom att lösa begynnelsevärdesproblemet

$$\begin{cases} x'(t) = \lambda(t) F_1(x(t), y(t)), & x(0) = x_0 \\ y'(t) = \lambda(t) F_2(x(t), y(t)), & y(0) = y_0 \end{cases}, \quad t \geq 0$$

för olika startpunkter (x_0, y_0).

Vi beräknar och ritar fältlinjer till $\boldsymbol{F}(x,y) = (\cos(x-y), \sin(xy))$ med `ode45` i MATLAB enligt

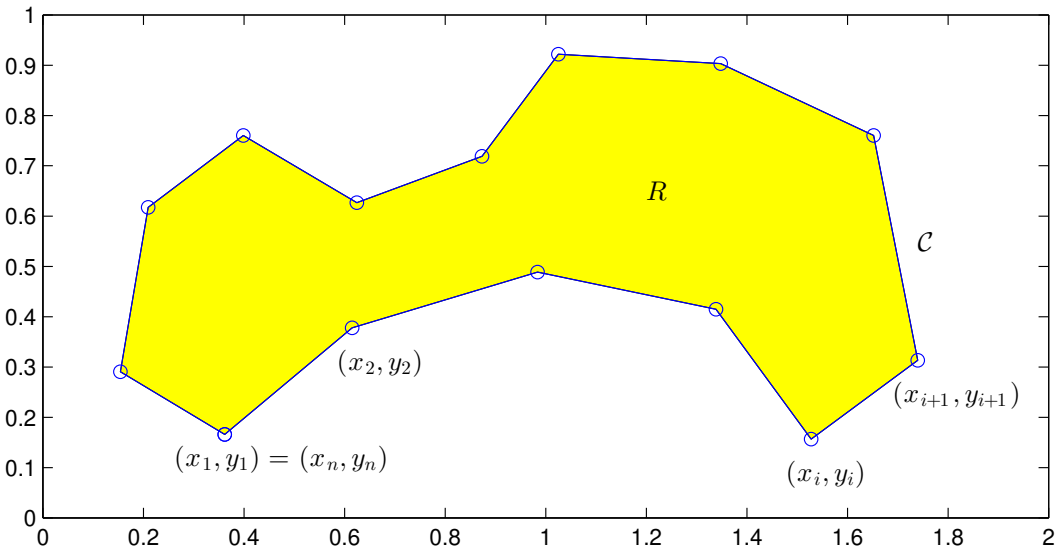
```
>> F=@(t,u)[cos(u(1)-u(2));sin(u(1).*u(2))];
>> hold on
>> [t,U]=ode45(F,[0 5],[0;1]);
>> plot(U(:,1),U(:,2),'g','linewidth',2)
>> [t,U]=ode45(F,[0 5],[0;0.5]);
>> plot(U(:,1),U(:,2),'g','linewidth',2)
>> hold off
```



Uppgift 4. Rita fält och fältlinjer till $\boldsymbol{F}(x,y) = (xy, x-y)$ över området $[-2,\,2] \times [-2,\,2]$.

5 Area av polygonområde med Greens formel

Vi tänker oss att vi har ett polygontåg $(x_1,y_1),(x_2,y_2)\cdots,(x_n,y_n)$ som inte korsar sig självt och som är slutet, dvs. att $x_n = x_1$ och $y_n = y_1$. Här är ett exempel som vi ritat en figur av



Om vi vill beräkna arean av det område R som omsluts av polygontåget kan vi använda formeln

$$A = \left| \frac{1}{2} \sum_{i=1}^{n-1} (x_{i+1} + x_i)(y_{i+1} - y_i) \right| \tag{1}$$

Så här beräknar vi arean i MATLAB

```
>> n=length(x);
>> A=0;
>> for i=1:n-1
       A=A+(x(i+1)+x(i))*(y(i+1)-y(i))/2;
   end
>> A=abs(A)
```

eller kortare

```
>> A=abs(sum((x(2:end)+x(1:end-1)).*(y(2:end)-y(1:end-1))/2))
```

Vi kan använda Greens formel för att bevisa formeln. Låt R beteckna området som omsluts av polygontåget och låt $\mathcal{C}$ beteckna områdets rand (med positiv orientering). Med $F_1 = 0$, $F_2 = x$ har vi $\frac{\partial F_1}{\partial y} = 0$ och $\frac{\partial F_2}{\partial x} = 1$ och därmed ger Greens formel

$$\text{Area}(R) = \iint_R 1 \, dx \, dy = \iint_R \left(\frac{\partial F_2}{\partial x} - \frac{\partial F_1}{\partial y} \right) dx \, dy =$$

$$= \oint_{\mathcal{C}} F_1(x, y) \, dx + F_2(x, y) \, dy = \oint_{\mathcal{C}} x \, dy$$

Nu gäller det att

$$\oint_{\mathcal{C}} x \, dy = \sum_{i=1}^{n-1} \int_{\mathcal{C}_i} x \, dy$$

där $\mathcal{C}_i$ är randsegmentet från (x_i, y_i) till (x_{i+1}, y_{i+1}).

Vi kan enkelt parametrisera randsegmenten

$$\begin{cases} x(t) = x_i + t\,(x_{i+1} - x_i) \\ y(t) = y_i + t\,(y_{i+1} - y_i) \end{cases} , \quad 0 \le t \le 1$$

Alltså har vi

$$\int_{\mathcal{C}_i} x(t) \, dy = \int_0^1 x(t) \, y'(t) \, dt =$$

$$= (y_{i+1} - y_i) \int_0^1 (x_i + t\,(x_{i+1} - x_i)) \, dt = (y_{i+1} - y_i) \frac{1}{2} (x_{i+1} + x_i)$$

Vi har slutligen kommit fram till

$$\text{Area}(R) = \sum_{i=1}^{n-1} \int_{\mathcal{C}_i} x \, dy = \frac{1}{2} \sum_{i=1}^{n-1} (y_{i+1} - y_i)(x_{i+1} + x_i)$$

Om randen $\mathcal{C}$ inte är positivt orienterad, får vi ta beloppet.

Uppgift 5. Använd areaformeln (1) på polygontågen från uppgift 3. Se om du får konvergens. Räkna sedan ut (för hand) ett exakt svar där du använder Greens formel på parametriseringen från uppgift 1.

6 Ytor i $\mathbb{R}^3$

Som exempel på en allmän yta i rummet tar vi en cylinder med radien ρ och höjden h som beskrivs av ekvationen

$$\{(x, y, z) : \ x^2 + y^2 = \rho^2, \ 0 \leq z \leq h\}$$

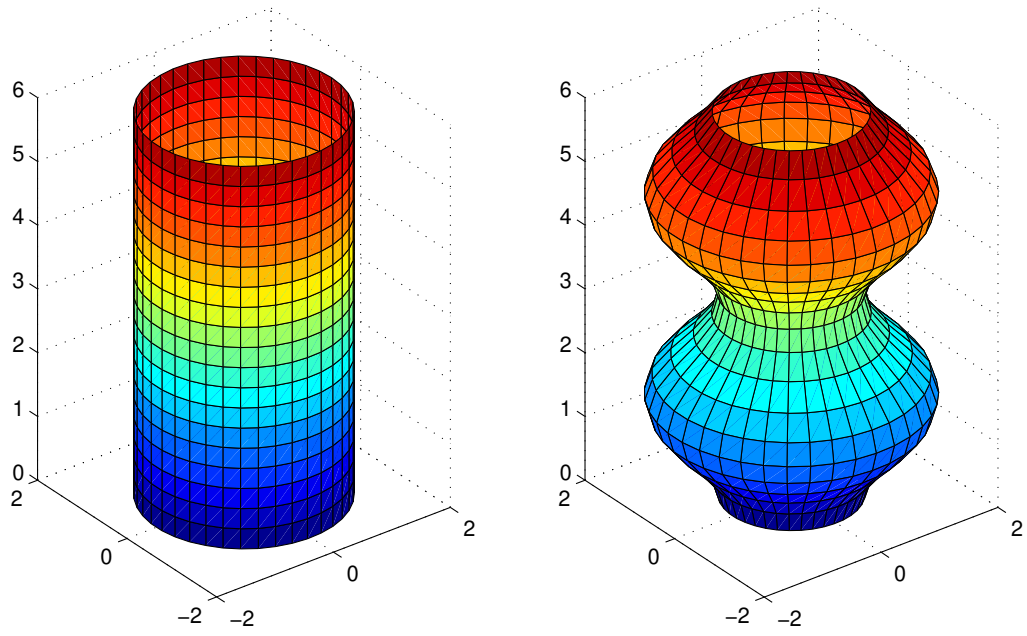
Ytan kan parametriseras $\boldsymbol{r} : \mathbb{R}^2 \to \mathbb{R}^3$ med

$$\boldsymbol{r}(s, t) = (x(s, t), y(s, t), z(s, t)) = (\rho \cos(t), \rho \sin(t), s)$$

där $0 \leq s \leq h$ och $0 \leq t \leq 2\pi$.

Detta passar bra när vi skall rita bilden i MATLAB.

```
>> rho=1.5; h=6; n=20; m=30;
>> s=linspace(0,h,n); t=linspace(0,2*pi,m);
>> [S,T]=meshgrid(s,t);
>> X=rho*cos(T); Y=rho*sin(T); Z=S;
>> surf(X,Y,Z)
>> axis equal, axis([-2 2 -2 2 0 6])
```



För cylindern hade vi $\rho(s) = \rho_0$, dvs. ett konstant värde (samma radie). Om vi istället låter $\rho(s) = 1 + \sin(s)^2$ (varierande radie) så blir det lite roligare. Detta är ett exempel på en *rotationsyta*. Sådana pratade vi om i envariabelkursen i samband med volym- och mantelareaberäkning.

Nu ritar vi upp rotationsytan. Var skiljer sig koden nedan från den för cylindern?

```
>> h=6; n=20; m=30;
>> s=linspace(0,h,n); t=linspace(0,2*pi,m);
>> [S,T]=meshgrid(s,t);
>> R=1+sin(S).^2;
>> X=R.*cos(T); Y=R.*sin(T); Z=S;
>> surf(X,Y,Z)
>> axis equal, axis([-2 2 -2 2 0 6])
```

Vi ser nu avslutningsvis på två parametriserade ytor som till utseendet är välbekanta.

En sfär med radien r och centrum i origo ges av ekvationen

$$x^2 + y^2 + z^2 = \rho^2$$

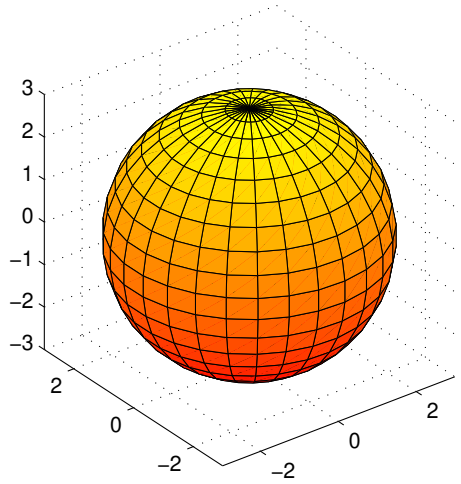
och kan parametriseras $\boldsymbol{r} : \mathbb{R}^2 \to \mathbb{R}^3$ med

$$\boldsymbol{r}(s,t) = (x(s,t), y(s,t), z(s,t)) = (\rho\sin(s)\cos(t), \rho\sin(s)\sin(t), \rho\cos(s))$$

där $0 \leq s \leq \pi$ och $0 \leq t \leq 2\pi$.

Vi ritar en sfär med radien $\rho = 3$ enligt

```
>> rho=3; n=20; m=30;
>> s=linspace(0,pi,n); t=linspace(0,2*pi,m);
>> [S,T]=meshgrid(s,t);
>> X=rho*sin(S).*cos(T); Y=rho*sin(S).*sin(T); Z=rho*cos(S);
>> surf(X,Y,Z)
>> axis equal
>> colormap(autumn)
```



En torus med lateralradien ρ och centralradien a samt centrum i origo ges av

$$(x^2 + y^2 + z^2 + a^2 - \rho^2)^2 = 4a^2(x^2 + y^2)$$

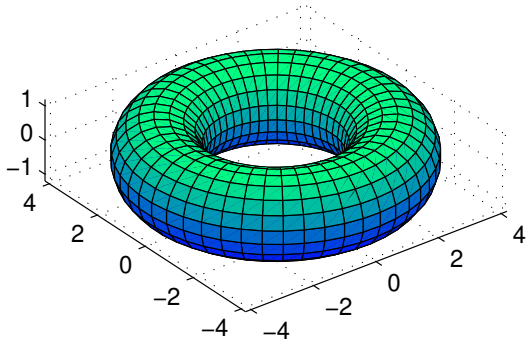
och kan parametriseras $\boldsymbol{r} : \mathbb{R}^2 \to \mathbb{R}^3$ med

$$\begin{aligned}\boldsymbol{r}(s,t) &= (x(s,t), y(s,t), z(s,t)) = \\ &= ((a + \rho\cos(s))\cos(t), (a + \rho\cos(s))\sin(t), \rho\sin(s))\end{aligned}$$

där $-\pi \leq s \leq \pi$ och $0 \leq t \leq 2\pi$.

Vi ritar en torus med lateralradien $\rho = 1.2$ och centralradien $a = 3$ enligt

```
>> rho=1.2; a=3; n=20; m=50;
>> s=linspace(-pi,pi,n); t=linspace(0,2*pi,m);
>> [S,T]=meshgrid(s,t);
>> X=(a+rho*cos(S)).*cos(T); Y=(a+rho*cos(S)).*sin(T); Z=rho*sin(S);
>> surf(X,Y,Z)
>> axis equal
>> colormap(winter)
```



Vi kan också lägga på belysning, välja mellan olika belysningsmodeller och välja mellan olika reflexionsmodeller (material).

```
>> shading interp                % flat, interp, faceted
>> camlight right                % left, right, headlight
>> lighting phong                % none, flat, phong, gouraud
>> material shiny                % shiny, dull, metal
```

Med kommandot `surf(X,Y,Z,'facealpha',0.7)` får vi ytan lite genomskinlig, där värdet vi ger `facealpha` avgör graden av genomskinlighet (0 för helt transparent och 1 för helt solid).

Uppgift 6. Rita nu några sfärer i rummet med olika radie och medelpunkt, lägg på lite belysning och rotera det hela några varv.

Vi kan rotera runt scenen genom att använda `camorbit` på olika sätt

```
>> axis equal vis3d
>> camlight left
>> for i=1:200
       camorbit(5,0)
       drawnow; pause(0.05)
   end
```

```
>> axis equal vis3d
>> h=camlight('left');
>> for i=1:200
       camorbit(5,0)
       camlight(h,'left')
       drawnow; pause(0.05)
   end
```

Pröva och skriv in koden. Alternativet till vänster lägger på belysning och sedan följer vi med kameran runt. I alternativet till höger följer även belysningskällan med kameran runt. Pausen väljer man så att det går lagom fort på datorn man använder.

1 Målsättning

Avsikten med denna laboration är att vi skall med MATLAB grafiskt studera parametriserade kurvor och ytor samt vektorfält.

2 Kommentarer och förklaringar

Vi kan också beräkna och rita fältlinjer med funktionen `streamline`, men vi får mycket noggrannare resultat med `ode45`. Skall vi rita vektorfält i rummet använder vi `quiver3`. Vi gjorde en egen parametrisering av en sfär, men det finns även en funktion `sphere` som genererar koordinatmatriser för en sfär.

3 Lärandemål

Efter denna laboration skall du kunna

- rita parametriserade kurvor i planet och rummet med `plot` och `plot3`
- rita vektorfält och strömlinjer med `quiver`, `quiver3` och `ode45`
- rita parametriserade ytor i rummet med `surf`