Idag: Mer om tillämpning av diagonalisering
* stabilitet
* komplexa eg. värden

Numerisk lösning av egenvärdes-problem 5.8

Obs: facit till "Fackverk" i VeckoPM 4.

System av ODE

$$\begin{cases} x'(t) = Ax \\ x(0) = b \end{cases} \qquad A = PDP^{-1}, \quad P = [v_1, \dots, v_n]$$

Vi löste detta i F14: $x = Py$

$$y_k(t) = c_k e^{\lambda_k t}$$

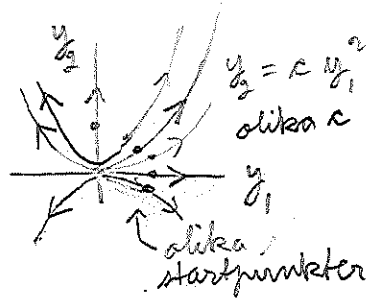
$$x(t) = Py(t) = c_1 e^{\lambda_1 t} v_1 + \dots c_n e^{\lambda_n t} v_n$$

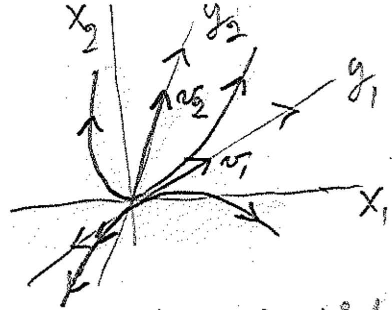
Formeln låter oss förstå.

Ex. $(n=2)$ * Båda $\lambda_1 > 0$, $\lambda_2 > 0$. T.ex. $\lambda_1 = 1$, $\lambda_2 = 2$

$$x(t) = c_1 e^{t} v_1 + c_2 e^{2t} v_2$$

$$y_2(t) = c_2 e^{2t} = \frac{c_2}{c_1^2}(c_1 e^{t})^2 = \frac{c_2}{c_1^2} y_1(t)^2 = c\, y_1(t)^2$$



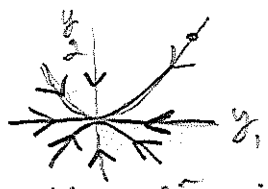


Instabilt system.

Alla lösningar går mot ∞ i både x_1 och x_2.

* Båda $\lambda_1 < 0$, $\lambda_2 < 0$. Ex. $\lambda_1 = -1$, $\lambda_2 = -2$.

På samma vis: $y_2(t) = c\, y_1(t)^2$

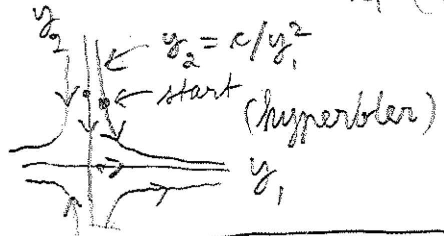


Stabilt system.

Alla lösningar går mot 0.

* En $\lambda_1 > 0$ och en $\lambda_2 < 0$. T.ex $\lambda_1 = 1$, $\lambda_2 = -2$.

$$y_2(t) = c_2 e^{-2t} = \frac{c_2}{c_1^2}\left(\frac{c_1}{e^{-t}}\right)^2 = c\frac{1}{y(t)^2}$$



Sadelpunkt

$\boxed{\text{Hyperboliskt system: instabilt.}}$

Förstå!! Inte räkna.

Ex Svängning.

$$u''(t) + u(t) = 0$$

$$\begin{cases} x_1 = u \\ x_2 = u' \end{cases} \Rightarrow \begin{cases} x_1' = u' = x_2 \\ x_2' = u'' = -u = -x_1 \end{cases}$$

Alltså: $x'(t) = Ax(t)$

med $A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$.

$$\det(A - \lambda I) = \begin{vmatrix} -\lambda & 1 \\ -1 & -\lambda \end{vmatrix} = \lambda^2 + 1, \quad \lambda_1 = i, \ \lambda_2 = -i$$

Med $\lambda = i$: $\left[\begin{array}{cc|c} -i & 1 & 0 \\ -1 & -i & 0 \end{array}\right] \sim \left[\begin{array}{cc|c} -i & 1 & 0 \\ 0 & 0 & 0 \end{array}\right] \sim \left[\begin{array}{ccc} 1 & i & 0 \\ 0 & 0 & 0 \end{array}\right]$

$x_2 = s$, $x_1 = -ix_2 = -is$

$$x = s\begin{bmatrix} -i \\ 1 \end{bmatrix} = s\begin{bmatrix} -i \\ -i^2 \end{bmatrix} = (-i)s\begin{bmatrix} 1 \\ i \end{bmatrix}, \quad v_1 = \begin{bmatrix} 1 \\ i \end{bmatrix}$$

Med $\lambda = -i$: $\left[\begin{array}{cc|c} i & 1 & 0 \\ -1 & i & 0 \end{array}\right] \sim \left[\begin{array}{cc|c} i & 1 & 0 \\ 0 & 0 & 0 \end{array}\right] \sim \left[\begin{array}{cc|c} 1 & -i & 0 \\ 0 & 0 & 0 \end{array}\right]$

$$x = s\begin{bmatrix} i \\ 1 \end{bmatrix} = is\begin{bmatrix} 1 \\ -i \end{bmatrix}, \quad v_2 = \begin{bmatrix} 1 \\ -i \end{bmatrix}$$

$$P = [v_1 \ v_2] = \begin{bmatrix} 1 & 1 \\ i & -i \end{bmatrix}, \quad \det P = 2i \neq 0$$

$$x(t) = c_1 e^{+it}\begin{bmatrix} 1 \\ +i \end{bmatrix} + c_2 e^{-it}\begin{bmatrix} 1 \\ -i \end{bmatrix}$$

$$= c_1(\cos t + i\sin t)\begin{bmatrix} 1 \\ +i \end{bmatrix} + c_2(\cos t - i\sin t)\begin{bmatrix} 1 \\ -i \end{bmatrix}$$

$$x_1(t) = c_1(\cos t + i\sin t) + c_2(\cos t - i\sin t)$$

$$= (c_1 + c_2)\cos t + (c_1 - c_2)i\sin t$$

$$= a\cos t + b\sin t \qquad \begin{cases} a = c_1 + c_2 \\ b = (c_1 - c_2)i \end{cases} \text{ reella!} \quad c_1, c_2 \text{ komplexa.}$$

$$x_2(t) = -(c_1 + c_2)\sin t + (c_1 - c_2)i\cos t$$

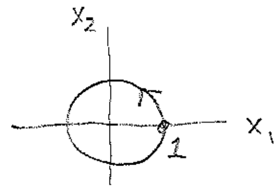
$$= -a\sin t + b\cos t \quad (= x_1'(t))$$

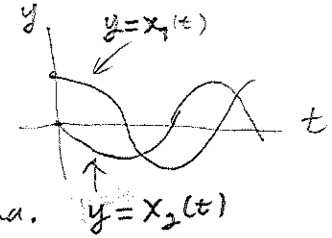
där a, b bestäms av begynnelsevillkor $x(0) = x_0$.

Obs: $x_1'(t) = x_2(t)$ som det ska vara

Svängning! T.ex. $a = 1$, $b = 0$ $x = \begin{bmatrix} \cos t \\ -\sin t \end{bmatrix}$

$x_1^2 + x_2^2 = \cos^2 t + \sin^2 t = 1$





Förstå! Inte räkna.

Mer allmänt:
$\lambda = \alpha + i\omega$
$e^{\lambda t} = e^{\alpha t} \cdot e^{i\omega t}$
$|e^{\lambda t}| = e^{\alpha t}$
$\alpha < 0 \Rightarrow$ stabil

Mer allmänt:

$$\lambda_k = \alpha_k + i\,\omega_k$$

$$e^{\lambda_k t} = e^{\alpha_k t} e^{i\omega_k t} = e^{\alpha_k t}\left(\cos(\omega_k t) + i \sin(\omega_k t)\right)$$

$$|e^{\lambda_k t}| = e^{\alpha_k t} \to 0 \text{ om } \alpha_k < 0.$$

<u>Stabilt</u> <u>system</u> om <u>alla</u> $\operatorname{Re} \lambda_k = \alpha_k < 0.$

Numerisk beräkning av egenvärden och egenvektorer.

Det går inte (utom för $n=2$, kanske $n=3$) att lösa $\det(A-\lambda I)=0$ exakt för det är en polynomekvation av grad n.

Vad göra? Approximera. Iterera.

Ide beräkna en följd av vektorer $x_1, x_2, \ldots \in \mathbb{R}^n$ som konvergerar mot en egenvektor v med $Av=\lambda v$.

Dvs $Ax_k \approx \lambda x_k$ och vi får en approximation av λ genom Rayleigh-kvoten $\dfrac{x_k^T A x_k}{x_k^T x_k} \approx \dfrac{v^T A v}{v^T v} =$

$$= \frac{v^T \lambda v}{v^T v} = \lambda .$$

Obs: $v^T v = |v|^2$, $v^T A v = v \cdot (Av)$

Den enklaste metoden är

Potensmetoden

Tag godtyckligt x och beräkna $x_k = A^k x$

dvs $x_1 = Ax$, $x_2 = Ax_1$, osv.

Om A har bas av egenvektorer $\{v_1, \ldots, v_n\}$ så har vi

$$x = c_1 v_1 + \ldots + c_n v_n \text{ (för några } c_k)$$

och

$$x_k = c_1 \lambda_1^k v_1 + \ldots + c_n \lambda_n^k v_n .$$

Om nu

$$|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|$$

strikt, dvs λ_1 är <u>dominant</u> egenvärde

så får vi

$$\frac{1}{\lambda_1^k} x_k = c_1 v_1 + c_2 \underbrace{\left(\frac{\lambda_2}{\lambda_1}\right)^k}_{\to 0} v_2 + \dots + c_n \underbrace{\left(\frac{\lambda_n}{\lambda_1}\right)^k}_{\to 0} v_n \to c_1 v$$

Koefficienterna $c_k\left(\frac{\lambda_k}{\lambda_1}\right)^k \to 0$ då $k \to \infty$ ty $\left|\frac{\lambda_k}{\lambda_1}\right| < 1$.

Alltså: $\frac{1}{\lambda_1^k} x_k \to c_1 v_1$, dvs en egenvektor hörande till λ_1.

(Måste ha $c_1 \neq 0$, annars ta ny startvektor x.)

Iterationen är

$$x_1 = \frac{1}{\lambda_1} x, \quad x_2 = \frac{1}{\lambda_1} x_2, \quad \text{och}$$

$$x_{k+1} = \frac{1}{\lambda_1} x_k$$

(skalad med $\frac{1}{\lambda_1}$ för att undvika $x_k \to \infty$)

I praktiken beräknar vi

$$x_{k+1} = \frac{1}{\frac{x_k^T A x_k}{x_k^T x_k}} x_k$$

eftersom vi inte vet λ_1.

Sedan får vi approx. av det <u>största egenvärdet</u>:

$$\lambda_1 \approx \frac{x_k A x_k}{x_k^T x_k}.$$

Kör Matlab demo, nästa sida

Metoden kan förfinas på olika sätt, men man får bara ett λ åt gången.

Andra iterativa metoder finns, t.ex. QR-metoden, som beräknar flera egenvektorer och egenvärden.

```
%% Konstruera A
% Diagonalisering:
P = [1 1; 1 -3];
D = [5 0; 0 1];

A = P*D/P % P*D*inv(P)

%% Räkna ut egenvektorer och egenvärden med MATLABs inbyggda metod:
[V, E] = eig(A)

%% Potensiteration
x = rand(2, 1)  % Slumpmässig startvektor

%% Gör detta om och om igen
x = A*x;
lambda = (x'*A*x) / (x'*x)  % Rayleigh-kvoten
x = x/lambda                % Skala om så att x inte blir oändligt stor
```