(59)

Věta: Implicitní numerická metoda může (ale nemusí) být A stabilní. Explicitní numerická metoda není nikdy A stabilní.

Absolutní stabilita pro lineární vícekrokovou metodu

Uvažujeme (LVM)

$$u^{n+1} = \sum_{j=0}^{p} a_j u^{n-j} + h \sum_{j=-1}^{p} b_j f^{n-j},$$

kde $f^{n-j} = \lambda u^{n-j}$

a uvažujeme řešení $u^n = \xi^{(n)}$ mocnina, tj.

$$\xi^{n+1} = \sum_{j=0}^{p} a_j \xi^{n-j} + \underbrace{\lambda h}_{=z} \sum_{j=-1}^{p} b_j \xi^{n-j} \quad \Big| \cdot \frac{1}{\xi^{n-p}}$$

$$\underbrace{\xi^{p+1} - \sum_{j=0}^{p} a_j \xi^{p-j}}_{\rho(\xi)} - z \underbrace{\sum_{j=-1}^{p} b_j \xi^{p-j}}_{\sigma(\xi)} = 0$$

60

$$\underline{\Pi(\xi, z) = \rho(\xi) - z\,\sigma(\xi) = 0}$$

$\rightarrow$ Kořeny rovnice $\Pi(\xi, z) = 0$ tvoří bázi řešení!

<u>Oblast absolutní stability (LVM)</u> je množina všech $z \in \mathbb{C}$, pro které platí, že kořeny ξ_j rovnice $\Pi(\xi, z) = 0$ splňují podmínky:

1) $|\xi_j| \leq 1$
2) je-li $|\xi_j| = 1$, pak je ξ_j jednoduchý kořen rovnice $\Pi(\xi, z) = 0$.

<u>Věta:</u> (Dahlquist)

1) Explicitní (LVM) nemůže být A stabilní.

2) A stabilní (LVM) má řád nejvýše 2.

(61)

Pozn. Pro určení oblasti absolutní stability (LVM) není možné postupně dosazovat všechna $z \in \mathbb{C}$ do $\pi(\xi, z) = 0$ a počítat kořeny ξ. Namísto toho hledáme všechna $z \in \mathbb{C}$, která leží na hranici Ω_a (tzv. Boundary locus method, viz LeVeque).

Na hranici Ω_a platí pro alespoň jeden kořen ξ_j vlastnost $|\xi_j| = 1$ tj.

$\xi_j = e^{i\vartheta}$, $\vartheta \in \langle 0; 2\pi)$. Kořen $\xi_j = e^{i\vartheta}$ dosadíme do rovnice $\pi(\xi, z) = 0$, tj.

$$\rho(e^{i\vartheta}) - z\,\sigma(e^{i\vartheta}) = 0$$

$$\boxed{z = \frac{\rho(e^{i\vartheta})}{\sigma(e^{i\vartheta})}, \quad \vartheta \in \langle 0; 2\pi)}$$

rovnice hranice Ω_a

62

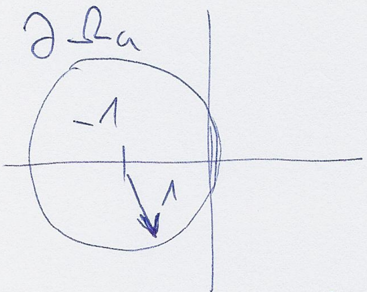
<u>Př.</u> "Boundary locus method" pro Eulerovu metodu

$$u^{n+1} = u^n + h f_n$$

$$u^{n+1} - u^n = h f_n$$

$$\Rightarrow \quad \rho(\xi) = \xi - 1 \quad \text{a} \quad \sigma(\xi) = 1$$

$$\Rightarrow \quad \underline{z} = \frac{\rho(e^{i\theta})}{\sigma(e^{i\theta})} = \underline{e^{i\theta} - 1}$$



hranice $\partial\Omega_a$ dělí komplexní rovinu na 2 části (uvnitř a vně kruhu).
Pro určení, která část komplexní roviny je Ω_a stačí dosadit náhodný bod např. z vnitřku kruhu (např. $z = -1$)

$$\Pi(\xi, z) = \rho(\xi) - z\,\sigma(\xi) = 0$$

$$\Pi(\xi, -1) = \rho(\xi) + \sigma(\xi) = \xi - 1 + 1 = 0 \quad \Rightarrow \quad \xi = 0$$

(63)

$\Rightarrow$ Sa je uvnitř kruhu tj.

Pozn: Příklady dalších oblastí viz LeVeque.

Pozn: Posouzení absolutní stability pro nelineární rovnici $u'(t) = f(t,u)$.

Uvažujeme $u(t) = u_0 + v(t)$, tj. po dosazení

$$\left(u_0 + v(t)\right)' = f(t, u_0 + v(t))$$

$$v'(t) = f(t, u_0 + v(t))$$

linearizace: $v'(t) = \underbrace{f(t, u_0)}_{\alpha} + \underbrace{\frac{\partial f}{\partial u}(t, u_0)}_{\lambda}\, v(t)$

rovnice $v'(t) = \lambda v(t) + \alpha$, λ, α jsou konst.

má analytické řešení $v(t) = C \cdot e^{\lambda t} - \dfrac{\alpha}{\lambda}$

tj. pro absolutní stabilitu je důležitá hodnota $\lambda \approx \frac{\partial f}{\partial u}(t_0, u_0)$.

Absolutní stabilita pro soustavy ODR

Uvažujme soustavu n $\underbrace{\text{obyčejných}}^{\text{lineárních}}$ diferenciálních rovnic 1. řádu ve tvaru

$$U' = A \cdot U ,$$

kde A je reálná matice typu $n \times n$

a $U = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_n(t) \end{bmatrix}$ je vektor neznámých.

Předpokládejme, že matici A lze zapsat jako

$$A = R \Lambda L , \quad \text{kde}$$

$R = \begin{bmatrix} | & | & & | \\ r_1 & r_2 & \cdots & r_n \\ | & | & & | \end{bmatrix}$ je matice pravostranných vlastních vektorů matice A

65

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & \\ 0 & \lambda_2 & 0 & \\ 0 & & \ddots & \\ & & & \lambda_n \end{bmatrix}$$ je diagonální matice vlastních čísel matice A

$$L = \begin{bmatrix} \text{———} & l_1 & \text{———} \\ \text{———} & l_2 & \text{———} \\ & \vdots & \\ \text{———} & l_n & \text{———} \end{bmatrix}$$ je matice levostranných vlastních vektorů matice A

kde
$$A r_i = \lambda_i r_i$$
$$l_i A = \lambda_i l_i \qquad i = 1, \dots, n$$

Lze ukázat že platí $AR = R\Lambda$
a $LA = \Lambda L$

$$\Rightarrow \quad A = R \Lambda R^{-1} = L^{-1} \Lambda L$$

$$\Rightarrow \quad \underline{R^{-1} = L}$$ neboli můžeme psát

$$\underline{A = R \Lambda L}$$

zpět k soustavě

$$U' = AU$$

$$L \cdot \mid U' = R \Lambda L U$$

$$LU' = \underbrace{LR}_{=E} \Lambda L U$$

$$(LU)' = \Lambda (LU)$$

Zavedeme substituci $V = LU = \begin{bmatrix} v_1(t) \\ v_2(t) \\ \vdots \\ v_n(t) \end{bmatrix}$

tj. $\underline{V' = \Lambda V}$ je vlastně

soustava n nezávislých rovnic

$$\underline{v_i'(t) = \lambda_i v_i(t) \quad , \quad i = 1, \dots, n}$$

Pro absolutní stabilitu musíme volit krok metody h tak, aby všechna $z_i = \lambda_i h$ pro $Re(\lambda_i) < 0$ ležela v oblasti absolutní stability použité metody. Zde již vidíme, že má smysl uvažovat λ komplexní (reálná matice

(67)

může mít komplexní vlastní čísla).

<u>Pozn:</u> Linearizace u soustavy nelineárních ODR

uvažujme soustavu ve tvaru

$$W'(t) = F(t, W(t)) \quad , \quad W(t_0) = W_0$$

kde $W(t) = \begin{bmatrix} w_1(t) \\ w_2(t) \\ \vdots \\ w_n(t) \end{bmatrix}$, $F = \begin{bmatrix} f_1(t, w_1, w_2, \dots, w_n) \\ f_2(\dots) \\ \vdots \\ f_n(\dots) \end{bmatrix}$

použijeme substituci

$$W(t) = W_0 + U(t)$$

tj:

$$U'(t) = F(t, W_0 + U(t))$$

po linearizaci

$$U'(t) = \underbrace{F(t, W_0)}_{=B} + \underbrace{\frac{\partial F}{\partial W}(t, W_0)}_{=A} \cdot U(t)$$

$$\underline{U'(t) = A\,U(t) + B}$$

(68)

<u>Př.:</u> Řešení CÚ

$$\varphi''(t) = -\alpha \sin\varphi(t), \quad \alpha = \frac{g}{L} > 0$$

$$\varphi(0)=0, \quad \varphi'(0)=1$$

(viz kyvadlo na str 1)

rovnici 2. řádu převedeme na soustavu 1. řádu

$$U' = F(t, U) \quad \text{kde}$$

$$\underline{U = \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}} = \begin{bmatrix} \varphi(t) \\ \varphi'(t) \end{bmatrix} \quad \text{a } F = \begin{bmatrix} u_2(t) \\ -\alpha \sin u_1(t) \end{bmatrix}$$

Pro absolutní stabilitu budeme potřebovat vlastní čísla matice

$$\frac{\partial F}{\partial U} = \begin{pmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \end{pmatrix} =$$

$$= \begin{pmatrix} 0 & 1 \\ -\alpha \cos u_1 & 0 \end{pmatrix}$$

vl. čísla $\frac{\partial F}{\partial U}$:

$$\begin{vmatrix} -\lambda & 1 \\ -\alpha \cos u_1 & -\lambda \end{vmatrix} = \lambda^2 + \alpha \cos u_1 = 0$$

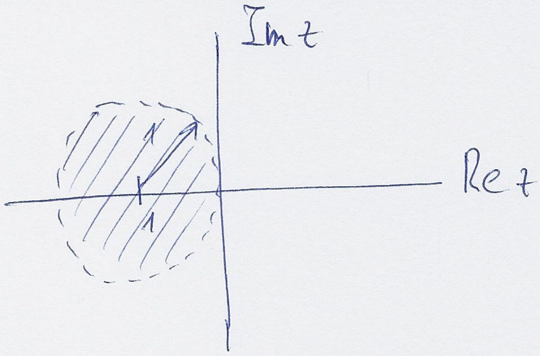
$$\lambda^2 = -\alpha \cos u_1$$

(69)

pro $\cos u_1 \in \langle 0, 1\rangle$ je $\lambda_{1,2} = \pm i\sqrt{\alpha \cos u_1}$

pro $\cos u_1 \in \langle -1, 0\rangle$ je $\lambda_{1,2} = \pm \sqrt{-\alpha \cos u_1}$

Budeme uvažovat explicitní Eulerovu metodu $U^{n+1} = U^n + h F_n$, která má oblast abs. stability



Pouze $\lambda = -\sqrt{-\alpha \cos u_1}$ splňuje $\operatorname{Re}\lambda < 0$.

Pro podmínku stability budeme uvažovat "nejhorší" případ, tj. $\cos u_1 = -1 \implies$

$\implies \lambda = -\sqrt{\alpha}$. Z oblasti abs. stability Eulerovy metody vidíme, že

$$-2 < \lambda h < 0$$

$$-2 < -\sqrt{\alpha}\, h < 0 \quad \Big| \cdot \left(-\frac{1}{\sqrt{\alpha}}\right)$$

$$\underline{\underline{\frac{2}{\sqrt{\alpha}} > h > 0}}$$

(70)

V souboru "kyvadlo.mws", který je možné spustit v programu Maple je realizován numerický výpočet Eulerovou metodou pro

poč. p. $u_1(0) = 1$

$u_2(0) = 1$

parametr $\lambda = 100$

počítá se řešení v čase $t = 10$

interval pro $t \in \langle 0; 10)$ se dělí na n dílů $\Rightarrow$ krok je $\frac{10}{n}$.

Jsou přiloženy výpočty pro

- $n = 48 \rightarrow$ krok $= 0{,}208 > \frac{2}{\sqrt{\underbrace{100}_{=\lambda}}} = 0{,}2$

  $\rightarrow$ tj. není splněna stabilita

- $n = 60 \rightarrow$ krok $= 0{,}166 < 0{,}2$ již stabilní, ale malá přesnost $\approx$ konst. $\cdot\, 0{,}166$ (tj. chyba okolo 20% i více)

- $n = 1000 \rightarrow$ krok $= 0{,}01 < 0{,}2$ stabilní a "přesné" (chyba okolo 1%)

71

Kyvadlo.mws

```
> restart:
> n:=48;dt:=10./n;alpha:=100;t:=0;
```

$$n := 48$$

$$dt := 0.2083333333$$

$$\alpha := 100$$

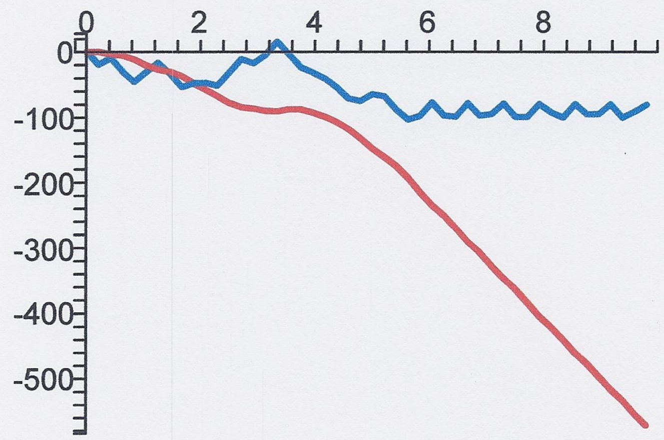
$$t := 0$$

```
> y1:=1;y2:=1;
```

$$yl := 1$$

$$y2 := 1$$

```
> y1num:=array(1..n):y2num:=array(1..n):
> y1num[1]:=(0,y1):y2num[1]:=(0,y2):
> for i from 2 to n do
> y1:=y1+dt*(y2):y2:=y2+dt*(-100*sin(y1)):
> t:=t+dt:
> y1num[i]:=(t,y1):y2num[i]:=(t,y2):
> end do:
> y1nump:=convert(y1num,listlist):y2nump:=convert(y2num,listlist):
> poloha:=pointplot(y1nump,symbol=circle,style=line,symbolsize=12,thickne
s=2,color=red):
> rychlost:=pointplot(y2nump,symbol=circle,style=line,symbolsize=12,thickn
ess=2,color=blue):
> with(plots):
Warning, the name changecoords has been redefined
> display(poloha,rychlost);
```



72

Kyvadlo.mws

```
> restart:
> n:=60;dt:=10./n;alpha:=100;t:=0;
```

$$n := 60$$

$$dt := 0.1666666667$$

$$\alpha := 100$$

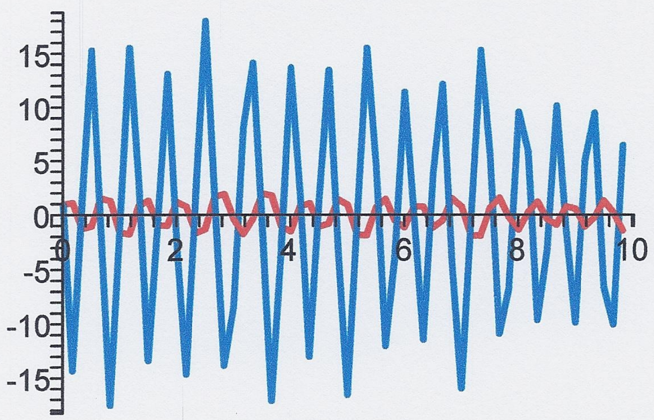
$$t := 0$$

```
> y1:=1;y2:=1;
```

$$y1 := 1$$

$$y2 := 1$$

```
> y1num:=array(1..n):y2num:=array(1..n):
> y1num[1]:=(0,y1):y2num[1]:=(0,y2):
> for i from 2 to n do
> y1:=y1+dt*(y2):y2:=y2+dt*(-100*sin(y1)):
> t:=t+dt:
> y1num[i]:=(t,y1):y2num[i]:=(t,y2):
> end do:
> y1nump:=convert(y1num,listlist):y2nump:=convert(y2num,listlist):
> poloha:=pointplot(y1nump,symbol=circle,style=line,symbolsize=12,thickne
s=2,color=red):
> rychlost:=pointplot(y2nump,symbol=circle,style=line,symbolsize=12,thickn
ess=2,color=blue):
> with(plots):
Warning, the name changecoords has been redefined
> display(poloha,rychlost);
```




```
>
```

(73)

kyvadlo.mws

```
> restart:
> n:=1000;dt:=10./n;alpha:=100;t:=0;
```

$$n := 1000$$

$$dt := 0.01000000000$$

$$\alpha := 100$$

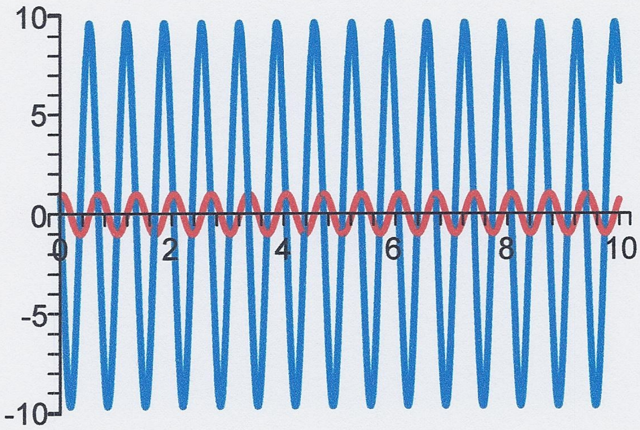
$$t := 0$$

```
> y1:=1;y2:=1;
```

$$y1 := 1$$

$$y2 := 1$$

```
> y1num:=array(1..n):y2num:=array(1..n):
> y1num[1]:=(0,y1):y2num[1]:=(0,y2):
> for i from 2 to n do
> y1:=y1+dt*(y2):y2:=y2+dt*(-100*sin(y1)):
> t:=t+dt:
> y1num[i]:=(t,y1):y2num[i]:=(t,y2):
> end do:
> y1nump:=convert(y1num,listlist):y2nump:=convert(y2num,listlist):
> poloha:=pointplot(y1nump,symbol=circle,style=line,symbolsize=12,thickne
s=2,color=red):
> rychlost:=pointplot(y2nump,symbol=circle,style=line,symbolsize=12,thick
ess=2,color=blue):
> with(plots):
Warning, the name changecoords has been redefined
> display(poloha,rychlost);
```




```
>
```

Poznámky k praktickému použití numerických metod

Určení (časového) kroku metody:

1) požadujeme dodržení určité přesnosti výsledku pro $t = T$, tj. $|u^N - u^*(T)| < \varepsilon$, kde $N = \frac{T}{h}$, h je krok metody. Je-li metoda řádu p, pak globální chyba po N-krocích je

$$N \cdot h^{p+1} \cdot \text{konst} = \underbrace{T \cdot \text{konst}\, h^p < \varepsilon}$$

$\Rightarrow$ uvažujeme-li $T h^p < \varepsilon$

$$h^p < \frac{\varepsilon}{T}$$

$$\underbrace{h < \sqrt[p]{\frac{\varepsilon}{T}}}$$

2) dodržení podmínek stability

3) volit krok s ohledem na vhodnou diskretizaci pravé strany tj. funkce f v rovnici $u' = f$.