Použití ustalovacích algoritmů k návrhu oscilátorů, základní druhy vysokofrekvenčních oscilátorů

Josef Dobeš

18. října 2021

Implicitní numerická integrace soustav obvodových rovnic

1. Časově vážené diference

System obvodových nelineárních diferenciálně-algebraických rovnic je obecně definován implicitní formě

$$\boldsymbol{f}\left(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), t\right) = \mathbf{0}.$$

Implicitní numerická integrace soustav obvodových rovnic

1. Časově vážené diference

System obvodových nelineárních diferenciálně-algebraických rovnic je obecně definován implicitní formě

$$f\left(x(t), \dot{x}(t), t\right) = \mathbf{0}. \tag{1}$$

Označme $x(t_n)$ symbolem x_n, $n = 1, \ldots$ a definujme zpětné časově-vážené diference (podle T. Rübner-Petersena, analýza stability podle A. I. Petrenka)

$$\delta^{(0)} x_n = x_n, \quad \delta^{(k)} x_n = \delta^{(k-1)} x_n - \alpha_n^{(k-1)}\, \delta^{(k-1)} x_{n-1}, \quad k = 1, \ldots, k_n + 2,$$

Časově vážené diference
Prediktor
Korektor
Procedura s ϵ-algoritmem
Necitlivost ϵ-algoritmu
Collpittsův oscilátor
Obvody s rozprostřeným...
– Zesilovač s . . .
– Oscilátor s . . .
Zpětnovazební . . .
Oscilátor s tunelovou . . .
Domovská stránka
Strana 2 z 30
Zpět
Celá obrazovka
Zavřít
Konec

Implicitní numerická integrace soustav obvodových rovnic

1. Časově vážené diference

System obvodových nelineárních diferenciálně-algebraických rovnic je obecně definován implicitní formě

$$\boldsymbol{f}\left(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), t\right) = \boldsymbol{0}. \tag{1}$$

Označme $\boldsymbol{x}(t_n)$ symbolem $\boldsymbol{x}_n$, $n = 1, \ldots$ a definujme zpětné časově-vážené diference (podle T. Rübner-Petersena, analýza stability podle A. I. Petrenka)

$$\delta^{(0)}\boldsymbol{x}_n = \boldsymbol{x}_n, \quad \delta^{(k)}\boldsymbol{x}_n = \delta^{(k-1)}\boldsymbol{x}_n - \alpha_n^{(k-1)}\,\delta^{(k-1)}\boldsymbol{x}_{n-1}, \quad k = 1, \ldots, k_n + 2,$$

kde k_n je řád interpolačního polynomu použitého v posledním integračním kroku a činitelé α_n jsou rovněž určeny rekurentním vztahem:

$$\alpha_n^{(0)} = 1, \quad \alpha_n^{(k)} = \alpha_n^{(k-1)} \frac{t_n - t_{n-k}}{t_{n-1} - t_{n-1-k}}, \quad k = 1, \ldots, k_n + 1.$$

2. Prediktor

Extrapolace obvodových proměnných do času t_{n+1} označené $\boldsymbol{x}_{n+1}^{(0)}$ lze provést dříve definovanými faktory $\alpha_{n+1}^{(\ldots)}$ a diferencemi $\delta^{(\ldots)}\boldsymbol{x}_n$ v následující explicitní formě:

$$\boldsymbol{x}_{n+1}^{(0)} = \alpha_{n+1}^{(0)}\,\delta^{(0)}\boldsymbol{x}_n + \alpha_{n+1}^{(1)}\,\delta^{(1)}\boldsymbol{x}_n + \cdots = \sum_{k=0}^{k_{n+1}} \alpha_{n+1}^{(k)}\,\delta^{(k)}\boldsymbol{x}_n.$$

(Lze ukázat, že jde o sofistikovanější formu Newtonova interpolačního mnohočlenu[1].)

2. Prediktor

Extrapolace obvodových proměnných do času t_{n+1} označené $\boldsymbol{x}_{n+1}^{(0)}$ lze provést dříve definovanými faktory $\alpha_{n+1}^{(\dots)}$ a diferencemi $\delta^{(\dots)}\boldsymbol{x}_n$ v následující explicitní formě:

$$\boldsymbol{x}_{n+1}^{(0)} = \alpha_{n+1}^{(0)}\,\delta^{(0)}\boldsymbol{x}_n + \alpha_{n+1}^{(1)}\,\delta^{(1)}\boldsymbol{x}_n + \cdots = \sum_{k=0}^{k_{n+1}} \alpha_{n+1}^{(k)}\,\delta^{(k)}\boldsymbol{x}_n.$$

(Lze ukázat, že jde o sofistikovanější formu Newtonova interpolačního mnohočlenu[1].)

Podobný vztah lze odvodit pro extrapolaci vektoru derivací podle času

$$\dot{\boldsymbol{x}}_{n+1}^{(0)} = \beta_{n+1}^{(0)}\delta^{(0)}\boldsymbol{x}_n + \beta_{n+1}^{(1)}\delta^{(1)}\boldsymbol{x}_n + \cdots = \sum_{k=0}^{k_{n+1}} \beta_{n+1}^{(k)}\delta^{(k)}\boldsymbol{x}_n,$$

2. Prediktor

Extrapolace obvodových proměnných do času t_{n+1} označené $\boldsymbol{x}_{n+1}^{(0)}$ lze provést dříve definovanými faktory $\alpha_{n+1}^{(\dots)}$ a diferencemi $\delta^{(\dots)}\boldsymbol{x}_n$ v následující explicitní formě:

$$\boldsymbol{x}_{n+1}^{(0)} = \alpha_{n+1}^{(0)}\,\delta^{(0)}\boldsymbol{x}_n + \alpha_{n+1}^{(1)}\,\delta^{(1)}\boldsymbol{x}_n + \cdots = \sum_{k=0}^{k_{n+1}} \alpha_{n+1}^{(k)}\,\delta^{(k)}\boldsymbol{x}_n.$$

(Lze ukázat, že jde o sofistikovanější formu Newtonova interpolačního mnohočlenu[1].)

Podobný vztah lze odvodit pro extrapolaci vektoru derivací podle času

$$\dot{\boldsymbol{x}}_{n+1}^{(0)} = \beta_{n+1}^{(0)}\delta^{(0)}\boldsymbol{x}_n + \beta_{n+1}^{(1)}\delta^{(1)}\boldsymbol{x}_n + \cdots = \sum_{k=0}^{k_{n+1}} \beta_{n+1}^{(k)}\delta^{(k)}\boldsymbol{x}_n,$$

kde faktory $\beta_{n+1}^{(\dots)}$ jsou opět dané rekurentní rovnicí, která také obsahuje dříve definované násobitele $\alpha_{n+1}^{(\dots)}$:

$$\beta_{n+1}^{(0)} = 0, \quad \beta_{n+1}^{(k)} = \frac{\alpha_{n+1}^{(k-1)} + (t_{n+1} - t_{n+1-k})\beta_{n+1}^{(k-1)}}{t_n - t_{n-k}}, \quad k = 1, \dots, k_{n+1}.$$

[1] J. Dobeš, Reliable CAD analyses of CMOS RF and microwave circuits using smoothed gate capacitance models, AEÜ–Int. Jour. Electr. Comm., no. 6, 2003.

3. Korektor

Finální hodnoty $\boldsymbol{x}_{n+1} := \boldsymbol{x}_{n+1}^{(j_{\max,n+1})}$ v čase t_{n+1} se získají iteračním procesem podobným Newtonově-Raphsonově methodě (${}^{i}x$ označuje i-tý prvek vektoru $\boldsymbol{x}$)

$$\left[\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)_{n+1}^{(j)} + \left(\frac{\partial \boldsymbol{f}}{\partial \dot{\boldsymbol{x}}}\right)_{n+1}^{(j)} \underbrace{\left(\frac{\mathrm{d}\,{}^{i}\dot{x}}{\mathrm{d}\,{}^{i}x}\right)_{n+1}}_{\gamma_{n+1}}\right] \Delta \boldsymbol{x}_{n+1}^{(j)} = -\boldsymbol{f}_{n+1}^{(j)}, \quad j = 0, \ldots, j_{\max,n+1} < j_{\max 1},$$

3. Korektor

Finální hodnoty $\boldsymbol{x}_{n+1} := \boldsymbol{x}_{n+1}^{(j_{\max,n+1})}$ v čase t_{n+1} se získají iteračním procesem podobným Newtonově-Raphsonově methodě (${}^{i}x$ označuje i-tý prvek vektoru $\boldsymbol{x}$)

$$\left[\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)_{n+1}^{(j)} + \left(\frac{\partial \boldsymbol{f}}{\partial \dot{\boldsymbol{x}}}\right)_{n+1}^{(j)} \underbrace{\left(\frac{\mathrm{d}\, {}^{i}\dot{x}}{\mathrm{d}\, {}^{i}x}\right)_{n+1}}_{\gamma_{n+1}}\right] \Delta \boldsymbol{x}_{n+1}^{(j)} = -\boldsymbol{f}_{n+1}^{(j)}, \quad j = 0, \ldots, j_{\max,n+1} < j_{\max 1},$$

$n = 0, \ldots$, tj. opakovaným řešením soustavy lineárních rovnic při aplikování implicitní formy aproximace derivací:

$$\dot{\boldsymbol{x}}_{n+1}^{(j)} = \lim_{t_{n+2} \to t_{n+1}} \frac{\boldsymbol{x}_{n+2}^{(j)} - \boldsymbol{x}_{n+1}}{t_{n+2} - t_{n+1}} = \sum_{k=1}^{k_{n+1}} \frac{1}{t_{n+1} - t_{n+1-k}} \, \delta^{(k)} \boldsymbol{x}_{n+1}^{(j)}$$

$$\Rightarrow \gamma_{n+1} = \sum_{k=1}^{k_{n+1}} \frac{1}{t_{n+1} - t_{n+1-k}} \; \forall\, {}^{i}x \in \boldsymbol{x}\,.$$

3. Korektor

Finální hodnoty $\boldsymbol{x}_{n+1} := \boldsymbol{x}_{n+1}^{(j_{\max,n+1})}$ v čase t_{n+1} se získají iteračním procesem podobným Newtonově-Raphsonově methodě (${}^{i}x$ označuje i-tý prvek vektoru $\boldsymbol{x}$)

$$\left[\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)_{n+1}^{(j)} + \left(\frac{\partial \boldsymbol{f}}{\partial \dot{\boldsymbol{x}}}\right)_{n+1}^{(j)} \underbrace{\left(\frac{\mathrm{d}\,{}^{i}\dot{x}}{\mathrm{d}\,{}^{i}x}\right)_{n+1}}_{\gamma_{n+1}}\right] \Delta \boldsymbol{x}_{n+1}^{(j)} = -\boldsymbol{f}_{n+1}^{(j)}, \quad j = 0, \ldots, j_{\max,n+1} < j_{\max 1},$$

$n = 0, \ldots$, tj. opakovaným řešením soustavy lineárních rovnic při aplikování implicitní formy aproximace derivací:

$$\dot{\boldsymbol{x}}_{n+1}^{(j)} = \lim_{t_{n+2}\to t_{n+1}} \frac{\boldsymbol{x}_{n+2}^{(j)} - \boldsymbol{x}_{n+1}}{t_{n+2} - t_{n+1}} = \sum_{k=1}^{k_{n+1}} \frac{1}{t_{n+1} - t_{n+1-k}}\, \delta^{(k)} \boldsymbol{x}_{n+1}^{(j)}$$

$$\Rightarrow \gamma_{n+1} = \sum_{k=1}^{k_{n+1}} \frac{1}{t_{n+1} - t_{n+1-k}}\ \forall\, {}^{i}x \in \boldsymbol{x}\,.$$

Vektory $\boldsymbol{x}_{n+1}^{(\ldots)}$ a $\dot{\boldsymbol{x}}_{n+1}^{(\ldots)}$ získají nové hodnoty po vyřešení soustavy lineárních rovnic korektoru:

$$\boldsymbol{x}_{n+1}^{(j+1)} = \boldsymbol{x}_{n+1}^{(j)} + \Delta \boldsymbol{x}_{n+1}^{(j)}, \quad \dot{\boldsymbol{x}}_{n+1}^{(j+1)} = \dot{\boldsymbol{x}}_{n+1}^{(j)} + \gamma_{n+1} \Delta \boldsymbol{x}_{n+1}^{(j)}.$$

K potlačení možné divergence je možné použít novou proceduru pro práci s diferencemi $\Delta\boldsymbol{x}_{n+1}^{(j)}$ během každé iterace:

$$
\begin{aligned}
&\texttt{if}\quad j = 0\quad \texttt{then}\\
&\qquad \boldsymbol{x}^* := \boldsymbol{x}_{n+1}^{(0)},\ \dot{\boldsymbol{x}}^* := \dot{\boldsymbol{x}}_{n+1}^{(0)},\\
&\qquad \Delta\boldsymbol{x}^* := \Delta\boldsymbol{x}_{n+1}^{(0)},\\
&\qquad \boldsymbol{f}^* := \boldsymbol{f}_{n+1}^{(0)},\ \text{a (první) iterace je akceptována,}
\end{aligned}
$$

K potlačení možné divergence je možné použít novou proceduru pro práci s diferencemi $\Delta \boldsymbol{x}_{n+1}^{(j)}$ během každé iterace:

$$
\begin{aligned}
&\texttt{if}\quad j=0\quad \texttt{then}\\
&\qquad \boldsymbol{x}^* := \boldsymbol{x}_{n+1}^{(0)},\ \dot{\boldsymbol{x}}^* := \dot{\boldsymbol{x}}_{n+1}^{(0)},\\
&\qquad \Delta\boldsymbol{x}^* := \Delta\boldsymbol{x}_{n+1}^{(0)},\\
&\qquad \boldsymbol{f}^* := \boldsymbol{f}_{n+1}^{(0)},\ \text{a (první) iterace je akceptována,}\\
&\texttt{else}\\
&\qquad \texttt{if}\quad \frac{1}{l}\sum_{i=1}^{l}\frac{\left|{}^{i}f_{n+1}^{(j)}\right|}{\left|{}^{i}f^*\right| + {}^{i}f_{\text{null}}} < 1 \quad \texttt{then}\\
&\qquad\qquad \boldsymbol{x}^* := \boldsymbol{x}_{n+1}^{(j)},\ \dot{\boldsymbol{x}}^* := \dot{\boldsymbol{x}}_{n+1}^{(j)},\\
&\qquad\qquad \Delta\boldsymbol{x}^* := \Delta\boldsymbol{x}_{n+1}^{(j)},\\
&\qquad\qquad \boldsymbol{f}^* := \boldsymbol{f}_{n+1}^{(j)},\ \text{a iterace je akceptována,}
\end{aligned}
$$

K potlačení možné divergence je možné použít novou proceduru pro práci s diferencemi $\Delta \boldsymbol{x}_{n+1}^{(j)}$ během každé iterace:

$$
\begin{array}{l}
\texttt{if} \quad j = 0 \quad \texttt{then} \\
\qquad \boldsymbol{x}^* := \boldsymbol{x}_{n+1}^{(0)},\ \dot{\boldsymbol{x}}^* := \dot{\boldsymbol{x}}_{n+1}^{(0)}, \\
\qquad \Delta \boldsymbol{x}^* := \Delta \boldsymbol{x}_{n+1}^{(0)}, \\
\qquad \boldsymbol{f}^* := \boldsymbol{f}_{n+1}^{(0)},\ \text{a (první) iterace je akceptována,} \\
\texttt{else} \\
\qquad \texttt{if} \quad \dfrac{1}{l} \displaystyle\sum_{i=1}^{l} \frac{\left| {}^i f_{n+1}^{(j)} \right|}{\left| {}^i f^* \right| + {}^i f_{\text{null}}} < 1 \quad \texttt{then} \\
\qquad\qquad \boldsymbol{x}^* := \boldsymbol{x}_{n+1}^{(j)},\ \dot{\boldsymbol{x}}^* := \dot{\boldsymbol{x}}_{n+1}^{(j)}, \\
\qquad\qquad \Delta \boldsymbol{x}^* := \Delta \boldsymbol{x}_{n+1}^{(j)}, \\
\qquad\qquad \boldsymbol{f}^* := \boldsymbol{f}_{n+1}^{(j)},\ \text{a iterace je akceptována,} \\
\qquad \texttt{else} \\
\qquad\qquad \Delta \boldsymbol{x}^* := \dfrac{\Delta \boldsymbol{x}^*}{2}, \\
\qquad\qquad \boldsymbol{x}_{n+1}^{(j)} := \boldsymbol{x}^*,\ \dot{\boldsymbol{x}}_{n+1}^{(j)} := \dot{\boldsymbol{x}}^*, \\
\qquad\qquad \Delta \boldsymbol{x}_{n+1}^{(j)} := \Delta \boldsymbol{x}^*,\ \text{a iterace je zamítnuta.}
\end{array}
$$

Ustalovací algoritmus

4. Procedura s ϵ-algoritmem

Pro implicitní systém diferenciálně-algebraických rovnic, problém výpočtu periodického ustáleného stavu může být jednoduše formulován jako řešení nonlineární symbolické rovnice

$$\boldsymbol{x}_{\text{steady}} = \mathcal{I}\left(\boldsymbol{x}_{\text{steady}}, t_0, t_0 + T_{\text{steady}}\right),$$

kde $\mathcal{I}\left(\boldsymbol{x}_{\text{initcond}}, t_0, t_0 + T_{\text{interval}}\right)$ symbolizuje hodnoty po numerickém řešení implicitního nelineárního systému diferenciálně-algebraických rovnic na intervalu T_{interval} při použití počáteční podmínky $\boldsymbol{x}_{\text{initcond}}$.

Ustalovací algoritmus

4. Procedura s ϵ-algoritmem

Pro implicitní systém diferenciálně-algebraických rovnic, problém výpočtu periodického ustáleného stavu může být jednoduše formulován jako řešení nonlineární symbolické rovnice

$$\boldsymbol{x}_{\text{steady}} = \mathcal{I}\left(\boldsymbol{x}_{\text{steady}}, t_0, t_0 + T_{\text{steady}}\right),$$

kde $\mathcal{I}\left(\boldsymbol{x}_{\text{initcond}}, t_0, t_0 + T_{\text{interval}}\right)$ symbolizuje hodnoty po numerickém řešení implicitního nelineárního systému diferenciálně-algebraických rovnic na intervalu T_{interval} při použití počáteční podmínky $\boldsymbol{x}_{\text{initcond}}$.

Místo (často velmi dlouhé) numerické integrace lze provést mnohem kratší integraci. Vzorky řešení jsou bezprostředně zaznamenány po každé z period. Tyto vzorky se stávají vstupem pro skalární ϵ-algoritmus, který je schopen odhadnout stav systému v budoucnosti. Výstup algoritmu se stane novou počáteční podmínkou pro (1) a celý proces se opakuje.

Ustalovací algoritmus

4. Procedura s ϵ-algoritmem

Pro implicitní systém diferenciálně-algebraických rovnic, problém výpočtu periodického ustáleného stavu může být jednoduše formulován jako řešení nonlineární symbolické rovnice

$$\boldsymbol{x}_{\text{steady}} = \mathcal{I}\left(\boldsymbol{x}_{\text{steady}}, t_0, t_0 + T_{\text{steady}}\right),$$

kde $\mathcal{I}\left(\boldsymbol{x}_{\text{initcond}}, t_0, t_0 + T_{\text{interval}}\right)$ symbolizuje hodnoty po numerickém řešení implicitního nelineárního systému diferenciálně-algebraických rovnic na intervalu T_{interval} při použití počáteční podmínky $\boldsymbol{x}_{\text{initcond}}$.

Místo (často velmi dlouhé) numerické integrace lze provést mnohem kratší integraci. Vzorky řešení jsou bezprostředně zaznamenány po každé z period. Tyto vzorky se stávají vstupem pro skalární ϵ-algoritmus, který je schopen odhadnout stav systému v budoucnosti. Výstup algoritmu se stane novou počáteční podmínkou pro (1) a celý proces se opakuje.

Počet period potřebný pro extrapolační smyčku závisí na počtu pomalu odeznívajících přechodných dějů. Tento počet lze redukovat (numerickou) filtrací – dolní propustí provedenou numerickou integrací

$$\boldsymbol{x}_j^{(0)} := \boldsymbol{x}_j\left(t_0 + \Delta t_{\text{extpol}}\right) = \int_{t_0}^{t_0+\Delta t_{\text{extpol}}} \mathcal{F}\left(\boldsymbol{x}(t), t\right) \mathrm{d}t, \quad j = 1, \ldots, j_{\max_{\text{epsalg}}}, \tag{2}$$

kde $j = 1, \ldots, j_{\max_{\text{epsalg}}}$ reprezentuje číslo iterace ϵ-algoritmu a $\dot{\boldsymbol{x}}(t) = \mathcal{F}\left(\boldsymbol{x}(t), t\right)$ symbolizuje vector numericky integrovaných funkcí.

Celá posloupnost vzorků je pak získána pokračující implicitní numerickou integrací

$$\boldsymbol{x}_j^{(k)} := \boldsymbol{x}_j \left(t_0 + \Delta t_{\mathrm{extpol}} + \textstyle\sum_{i=1}^{k} T_j^{(i)} \right) = \int_{t_0+\Delta t_{\mathrm{extpol}}}^{t_0+\Delta t_{\mathrm{extpol}}+\sum_{i=1}^{k} T_j^{(i)}} \mathcal{F}\left(\boldsymbol{x}(t), t \right) \mathrm{d}t, \tag{3}$$

kde $j = 1, \ldots, j_{\mathrm{max}_{\mathrm{epsalg}}}$, $k = 1, \ldots, 2k_{\mathrm{extpol}}$ a $T_j^{(i)}$ označuje periody, které musí být pro autonomní obvody (např. oscilátory) určeny iteracemi (4).

Celá posloupnost vzorků je pak získána pokračující implicitní numerickou integrací

$$\boldsymbol{x}_j^{(k)} := \boldsymbol{x}_j\left(t_0 + \Delta t_{\text{extpol}} + \sum_{i=1}^{k} T_j^{(i)}\right) = \int_{t_0+\Delta t_{\text{extpol}}}^{t_0+\Delta t_{\text{extpol}}+\sum_{i=1}^{k} T_j^{(i)}} \mathcal{F}\left(\boldsymbol{x}(t), t\right) \mathrm{d}t, \tag{3}$$

kde $j = 1, \ldots, j_{\max_{\text{epsalg}}}$, $k = 1, \ldots, 2k_{\text{extpol}}$ a $T_j^{(i)}$ označuje periody, které musí být pro autonomní obvody (např. oscilátory) určeny iteracemi (4).

4.1. Scalární ϵ-algoritmus

Po získání všech hodnot vypočtených procesy (2) a (3), ϵ-algoritmus se inicializuje vztahy

$$\begin{aligned} {}^{i}\epsilon_{-1}^{(k)} &:= 0, & k &= 1, \ldots, 2k_{\text{extpol}}, \\ {}^{i}\epsilon_{0}^{(k)} &:= {}^{i}x_j^{(k)}, & k &= 0, \ldots, 2k_{\text{extpol}}, \end{aligned} \qquad i = 1, \ldots, l,$$

Celá posloupnost vzorků je pak získána pokračující implicitní numerickou integrací

$$\boldsymbol{x}_j^{(k)} := \boldsymbol{x}_j \left(t_0 + \Delta t_{\mathrm{extpol}} + \textstyle\sum_{i=1}^{k} T_j^{(i)}\right) = \int_{t_0+\Delta t_{\mathrm{extpol}}}^{t_0+\Delta t_{\mathrm{extpol}}+\sum_{i=1}^{k} T_j^{(i)}} \mathcal{F}\left(\boldsymbol{x}(t), t\right) \mathrm{d}t, \tag{3}$$

kde $j = 1, \ldots, j_{\max_{\mathrm{epsalg}}}$, $k = 1, \ldots, 2k_{\mathrm{extpol}}$ a $T_j^{(i)}$ označuje periody, které musí být pro autonomní obvody (např. oscilátory) určeny iteracemi (4).

4.1. Scalární ϵ-algoritmus

Po získání všech hodnot vypočtených procesy (2) a (3), ϵ-algoritmus se inicializuje vztahy

$$\left.\begin{array}{lll} {}^{i}\epsilon_{-1}^{(k)} := 0, & & k = 1, \ldots, 2k_{\mathrm{extpol}}, \\ {}^{i}\epsilon_{0}^{(k)} := {}^{i}x_j^{(k)}, & & k = 0, \ldots, 2k_{\mathrm{extpol}}, \end{array}\right\} \quad i = 1, \ldots, l,$$

a proces extrapolace se pak provede rekurentními vztahy:

$${}^{i}\epsilon_{m+1}^{(k)} := {}^{i}\epsilon_{m-1}^{(k+1)} + \frac{1}{{}^{i}\epsilon_{m}^{(k+1)} - {}^{i}\epsilon_{m}^{(k)}}, \; m = 0, \ldots, 2k_{\mathrm{extpol}} - 1, \; k = 0, \ldots, 2k_{\mathrm{extpol}} - 1 - m, \; i = 1, \ldots, l,$$

tj. pro celý vektor $\boldsymbol{\epsilon}_{m+1}^{(k)}$. Výsledek ϵ-algoritmu se stane novou počáteční podmínkou pro (1):

$$\boldsymbol{x}_{j+1}(t_0) := \boldsymbol{\epsilon}_{2k_{\mathrm{extpol}}}^{(0)}.$$

Evoluci ϵ-algoritmu lze znázornit následujícím diagramem (hodnoty v rozích trojúhelníků vytvářejí postupně nové hodnoty ve směru šipek):

$$
\begin{array}{ccccccc}
 & & {}^{i}\epsilon_{0}^{(0)} & & & & \\
{}^{i}\epsilon_{-1}^{(1)} & \triangleleft & & \longrightarrow & {}^{i}\epsilon_{1}^{(0)} & & \\
 & & {}^{i}\epsilon_{0}^{(1)} & \triangleleft & & \longrightarrow & {}^{i}\epsilon_{2}^{(0)} \\
{}^{i}\epsilon_{-1}^{(2)} & \triangleleft & & \longrightarrow & {}^{i}\epsilon_{1}^{(1)} & & \\
 & & {}^{i}\epsilon_{0}^{(2)} & & & & \dots \\
\dots & & & & \dots & & \\
 & & \dots & & & &
\end{array}
$$

Evoluci ϵ-algoritmu lze znázornit následujícím diagramem (hodnoty v rozích trojúhelníků vytvářejí postupně nové hodnoty ve směru šipek):

$$
\begin{array}{ccccccc}
 & & {}^{i}\epsilon_{0}^{(0)} & & & & \\
{}^{i}\epsilon_{-1}^{(1)} & \triangleleft & & \longrightarrow & {}^{i}\epsilon_{1}^{(0)} & & \\
 & & {}^{i}\epsilon_{0}^{(1)} & \triangleleft & & \longrightarrow & {}^{i}\epsilon_{2}^{(0)} \\
{}^{i}\epsilon_{-1}^{(2)} & \triangleleft & & \longrightarrow & {}^{i}\epsilon_{1}^{(1)} & & \\
 & & {}^{i}\epsilon_{0}^{(2)} & & & & \dots \\
\dots & & & & \dots & & \\
 & & \dots & & & &
\end{array}
$$

Procedura se opakuje, dokud není detekována konvergence($\epsilon_{\mathrm{extpol}}$ je povolená extrapolační chyba):

$$
\texttt{if} \quad \max_{i=1,\dots,l} \frac{\left| {}^{i}x_{j}^{(k)} - {}^{i}x_{j}^{(k-1)} \right|}{\left| {}^{i}x_{j}^{(k)} \right| + {}^{i}x_{\mathrm{null}}} \leqq \epsilon_{\mathrm{extpol}} \quad \texttt{then} \quad \boldsymbol{x}_{\mathrm{steady}} := \boldsymbol{x}_{j}^{(k)}, \quad k \in \langle 1, \dots, 2k_{\mathrm{extpol}} \rangle .
$$

Evoluci ϵ-algoritmu lze znázornit následujícím diagramem (hodnoty v rozích trojúhelníků vytvářejí postupně nové hodnoty ve směru šipek):

$$
\begin{array}{ccccccc}
 & & {}^{i}\epsilon_{0}^{(0)} & & & & \\
{}^{i}\epsilon_{-1}^{(1)} & \triangleleft & & \longrightarrow & {}^{i}\epsilon_{1}^{(0)} & & \\
 & & {}^{i}\epsilon_{0}^{(1)} & \triangleleft & & \longrightarrow & {}^{i}\epsilon_{2}^{(0)} \\
{}^{i}\epsilon_{-1}^{(2)} & \triangleleft & & \longrightarrow & {}^{i}\epsilon_{1}^{(1)} & & \\
 & & {}^{i}\epsilon_{0}^{(2)} & & & & \dots \\
\dots & & & & \dots & & \\
 & & \dots & & & &
\end{array}
$$

Procedura se opakuje, dokud není detekována konvergence(ϵ_{extpol} je povolená extrapolační chyba):

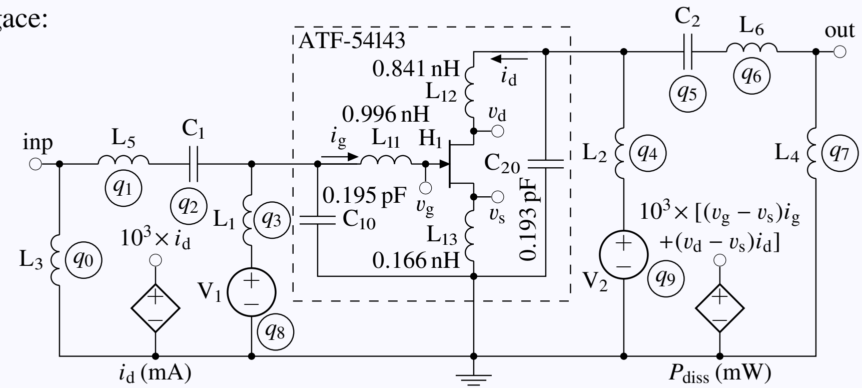
$$
\texttt{if} \quad \max_{i=1,\dots,l} \frac{\left| {}^{i}x_j^{(k)} - {}^{i}x_j^{(k-1)} \right|}{\left| {}^{i}x_j^{(k)} \right| + {}^{i}x_{\text{null}}} \leqq \epsilon_{\text{extpol}} \quad \texttt{then} \quad \boldsymbol{x}_{\text{steady}} := \boldsymbol{x}_j^{(k)}, \quad k \in \langle 1, \dots, 2k_{\text{extpol}} \rangle.
$$

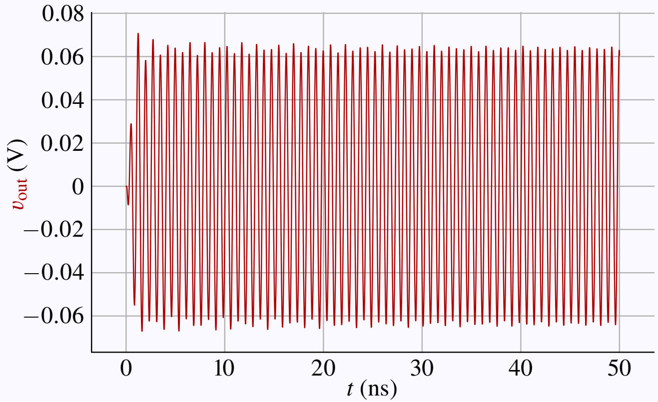
Určení period autonomních systémů se realizuje nalezením průsečíků vhodně vybrané (i_{fix})-té složky $\boldsymbol{x}_j$ s vhodnou (a realisticky zvolenou!) hodnotou ${}^{i_{\text{fix}}}x_j = x_{\text{fix}},\ i_{\text{fix}} \in \langle 1, l \rangle$:

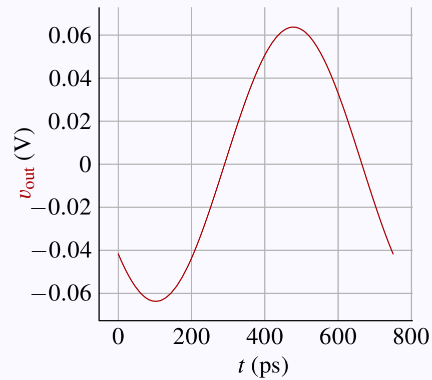
$$
{}^{i_{\text{fix}}}\dot{x}_j\left(t_{\text{period}}^{(\ell)}\right)\Delta t_{\text{period}}^{(\ell)} = x_{\text{fix}} - {}^{i_{\text{fix}}}x_j\left(t_{\text{period}}^{(\ell)}\right), \quad t_{\text{period}}^{(\ell+1)} = t_{\text{period}}^{(\ell)} + \Delta t_{\text{period}}^{(\ell)}, \quad \ell = 1, \dots, \ell_{\max}. \tag{4}
$$

5. Necitlivost ϵ-algoritmu

Necitlivost k řádu algoritmu lze ukázat na nízkošumovém zesilovači pro multi-konstelační přijímač satelitní navigace:







Domovská stránka
Strana 9 z 30
Zpět
Celá obrazovka
Zavřít
Konec

5.1. Porovnání nezbytného počtu integračních kroků pro získání ustáleného stavu

Utilized method	Used integration steps for interpolation orders						
	1st	2nd	3rd	4th	5th	6th	$\sum$
1st iteration	3	1	69	170	232	167	642
2nd iteration	5	7	69	147	243	113	584
3rd iteration	6	7	82	167	238	67	567
4th iteration	6	6	71	167	237	74	561
5th iteration	7	5	68	161	227	78	546
6th iteration	7	7	63	125	138	43	383
$\boldsymbol{\epsilon}$: $k_{\text{extpol}} = \mathbf{2}$	34	33	422	937	1315	542	**3283**
1st iteration	3	1	91	219	315	191	820
2nd iteration	7	9	109	226	290	118	759
3rd iteration	7	6	80	219	294	93	699
4th iteration	7	5	125	256	286	83	762
5th iteration	6	6	51	95	95	41	294
$\boldsymbol{\epsilon}$: $k_{\text{extpol}} = \mathbf{3}$	30	27	456	1015	1280	526	**3334**
1st iteration	3	1	127	281	390	213	1015
2nd iteration	7	7	123	253	229	84	703
$\boldsymbol{\epsilon}$: $k_{\text{extpol}} = \mathbf{4}$	10	8	250	534	619	297	**1718**
1st iteration	3	1	150	333	465	239	1191
2nd iteration	7	7	150	327	446	165	1102
$\boldsymbol{\epsilon}$: $k_{\text{extpol}} = \mathbf{5}$	10	8	300	660	911	404	**2293**

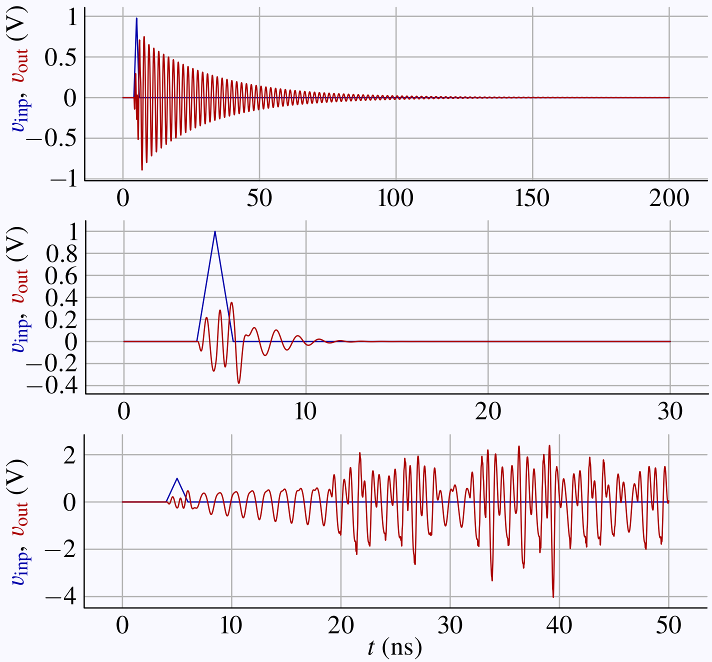
Porovnání nezbytného počtu integračních kroků pro získání ustáleného stavu (pokrač.)

Utilized method	Used integration steps for interpolation orders						
	1st	2nd	3rd	4th	5th	6th	$\sum$
1st iteration	3	1	173	373	549	263	1362
2nd iteration	7	7	112	229	249	88	692
$\boldsymbol{\epsilon}$: $k_{\text{extpol}} = \mathbf{6}$	10	8	285	602	798	351	**2054**
1st iteration	3	1	194	424	629	289	1540
2nd iteration	7	7	81	175	240	73	583
$\boldsymbol{\epsilon}$: $k_{\text{extpol}} = \mathbf{7}$	10	8	275	599	869	362	**2123**
1st iteration	3	1	221	468	711	314	1718
2nd iteration	7	7	92	216	259	88	669
$\boldsymbol{\epsilon}$: $k_{\text{extpol}} = \mathbf{8}$	10	8	313	684	970	402	**2387**
1st iteration	3	1	234	512	790	345	1885
2nd iteration	7	7	92	216	259	88	669
$\boldsymbol{\epsilon}$: $k_{\text{extpol}} = \mathbf{9}$	10	8	326	728	1049	433	**2554**
Transient	3	1	3630	8007	10741	3461	**25843**

Jak je ukázáno, bylo provedeno 25843 integračních kroků v případě standardní implicitní numerické integrace. Nicméně, pouze 1718 integračních kroků bylo provedeno v případě ϵ-algoritmu čtvrtého řádu (a podobné počty pro další řády interpolace, což potvrzuje necitlivost algoritmu vzhledem k jeho řádu).

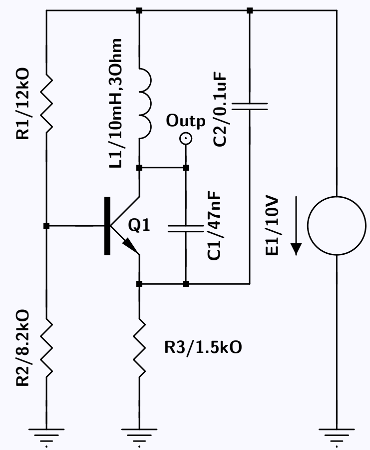
5.2. Klasifikace obvodových přechodných dějů z hlediska použitelnosti ϵ-algoritmu

Different responses of the amplifier to the (identical) triangular pulse: long and short transients or chaotic oscillations for three points of Pareto front. **The ϵ-algorithm is efficient in the first case, inefficient in the second case (usable, but not too faster than other methods), and unusable in the third case.**



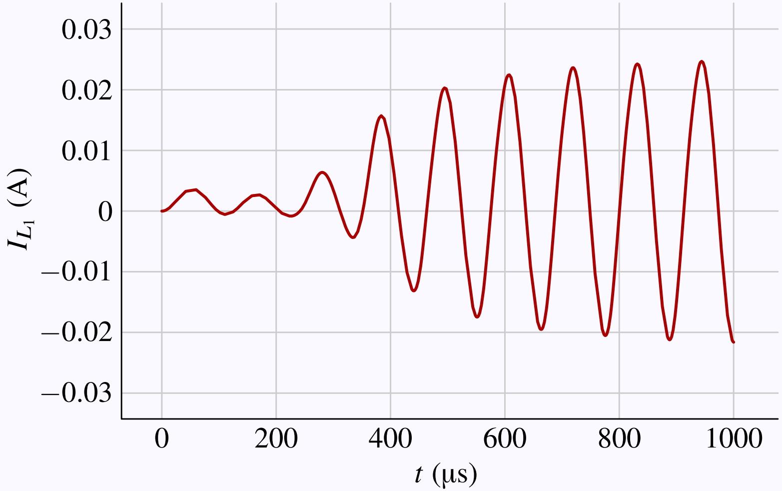
6. Collpittsův oscilátor

Simulací – časovou analýzou a ustalovacím algoritmem – ověřené zapojení Colpittsova oscilátoru:

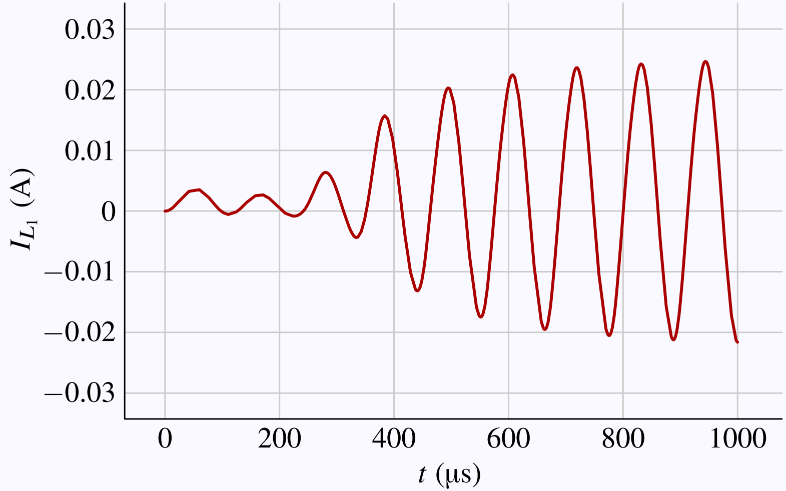


Použitá metoda	Počty integračních kroků						
	1.	2.	3.	4.	5.	6.	Celk.
1. iterace	22	88	171	182	126	108	697
2. iterace	19	50	98	91	57	35	350
Extrapol.	41	138	269	273	183	143	**1047**
Klasická	32	159	316	339	219	176	**1241**

6.1. Detail přechodného jevu - proud cívkou



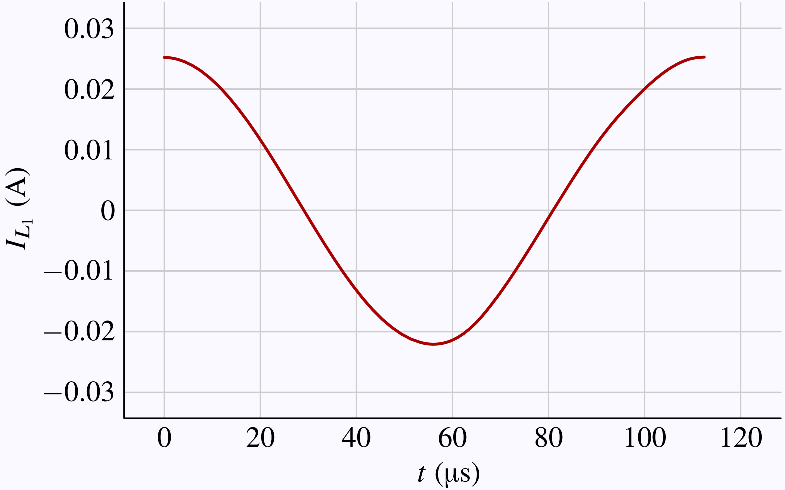
6.1. Detail přechodného jevu - proud cívkou



V porovnání s jinými oscilátory (krystalovými apod.) je zde poměrně krátký přechodný děj.

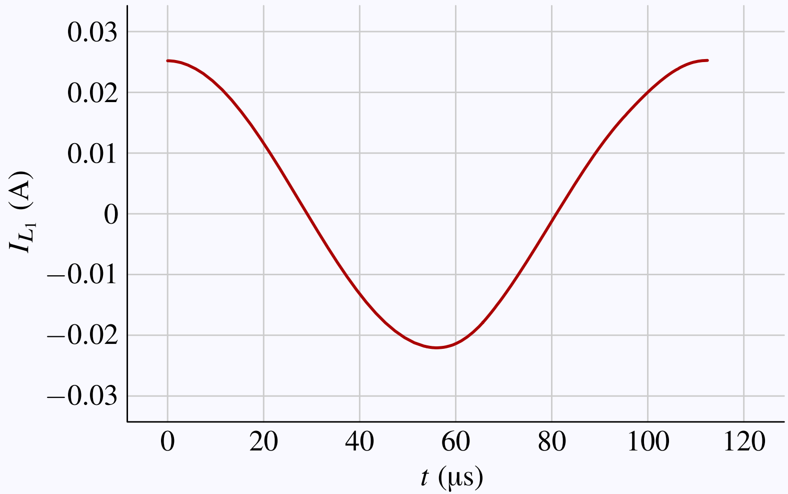
6.2. Ustálená periodická odezva

6.2.1. Proud cívkou



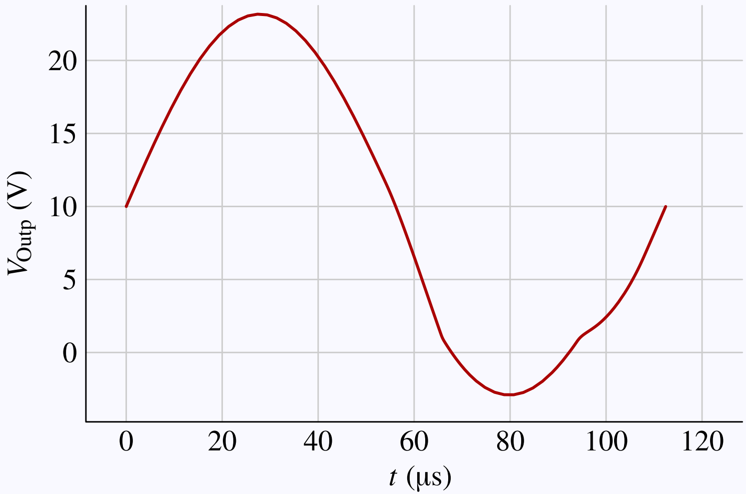
6.2. Ustálená periodická odezva

6.2.1. Proud cívkou

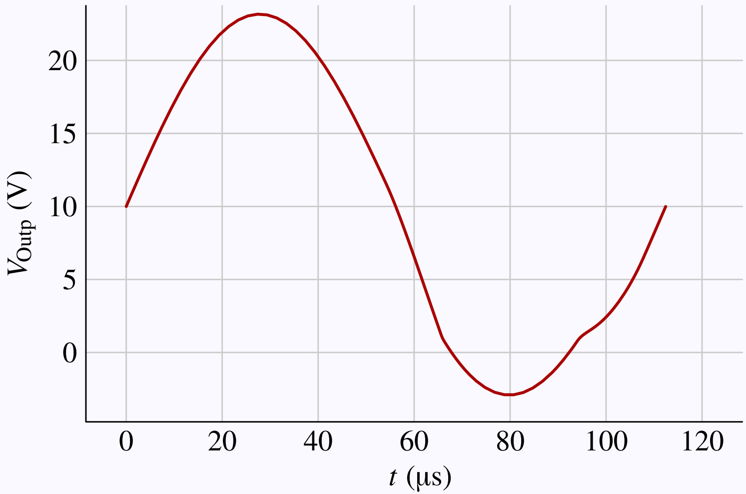


Koeficient harmonického zkreslení: 3.3 %

6.2.2. Výstupní napětí



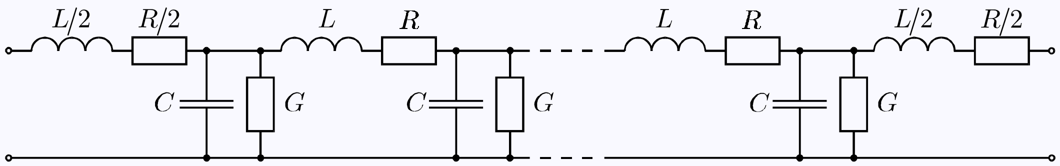
6.2.2. Výstupní napětí



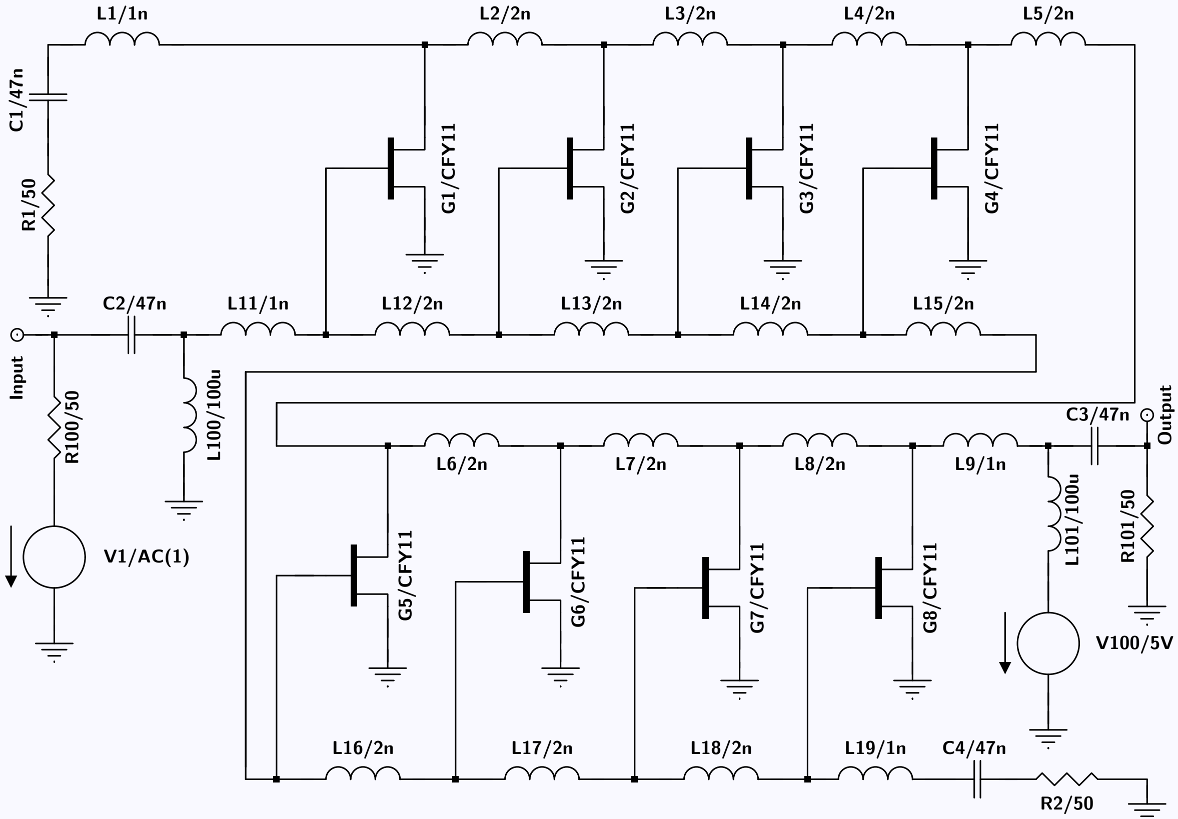
Koeficient harmonického zkreslení: 7.9 % (!)

7. Obvody s rozprostřeným zesílením

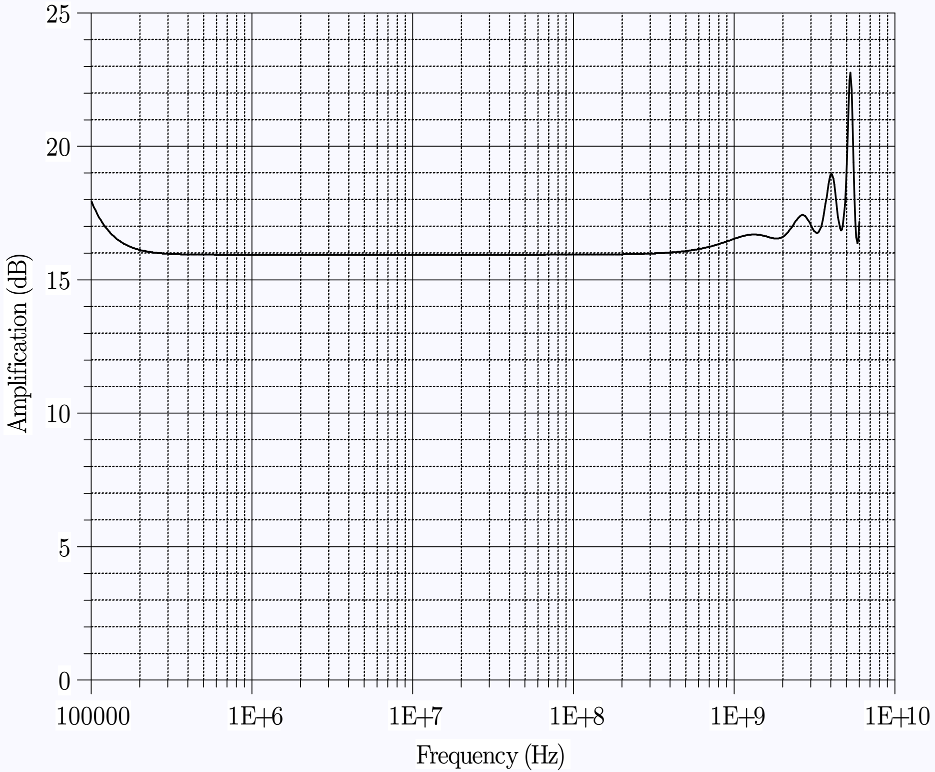
Využívají „simulace“ přenosového vedení:



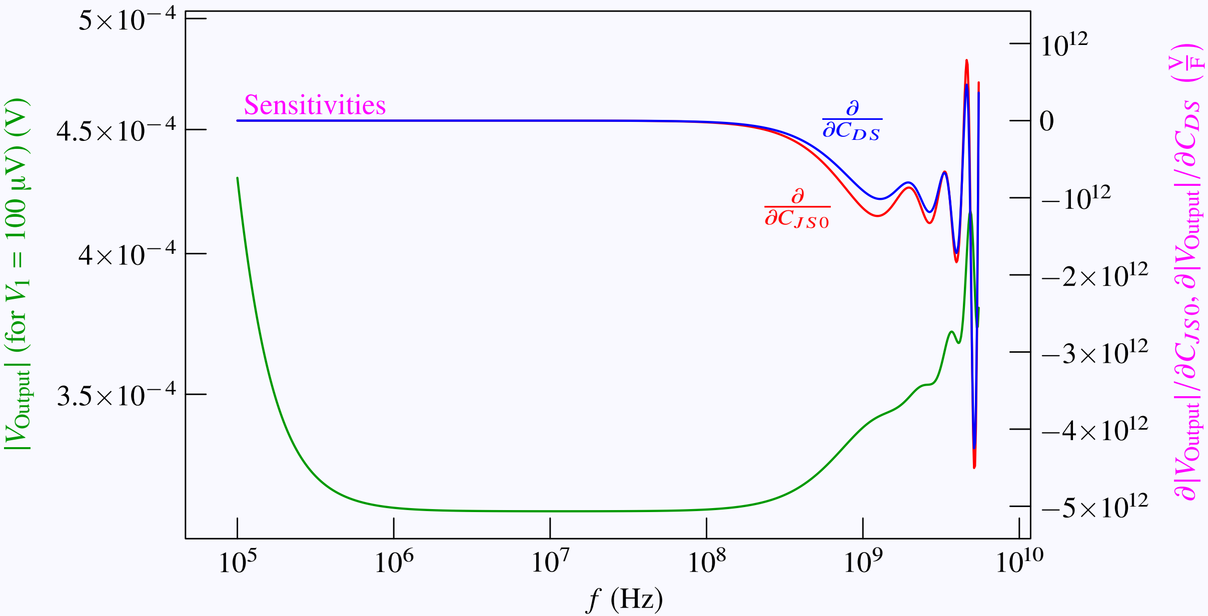
7.1. Zesilovač s rozprostřeným zesílením



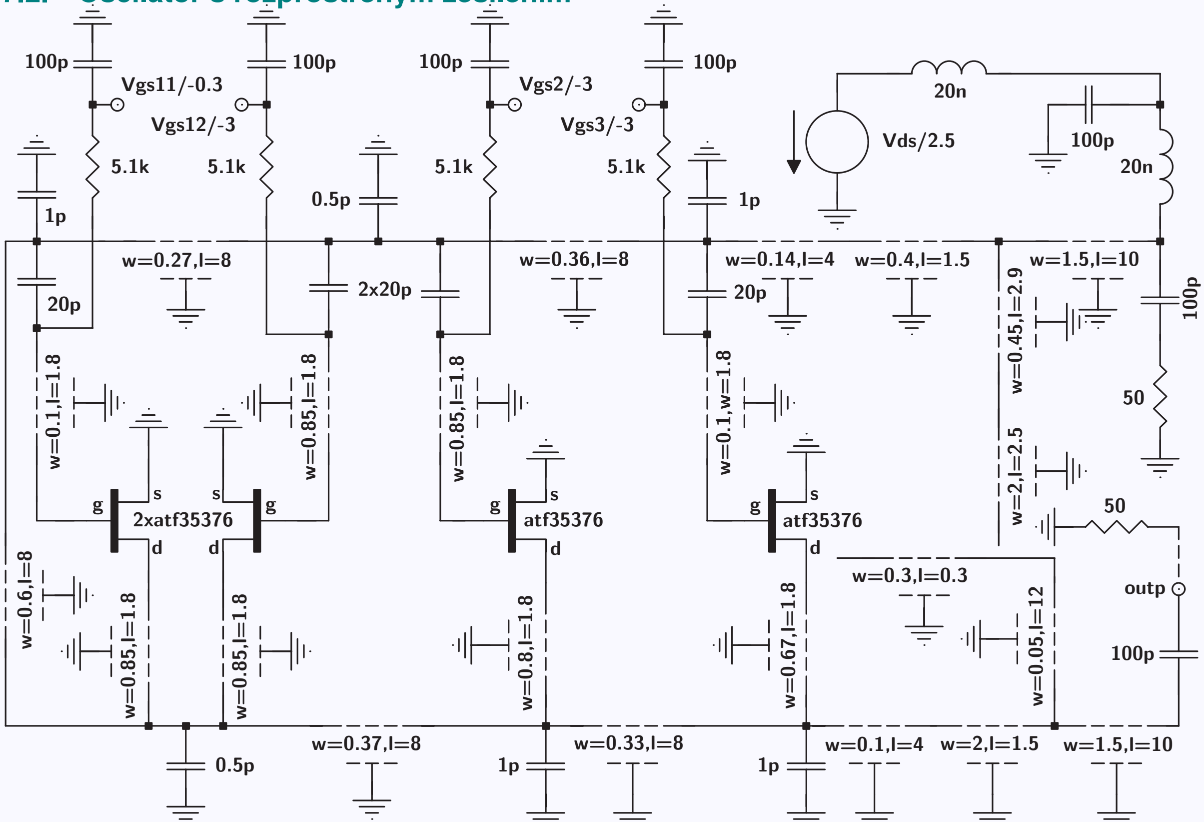
7.1.1. Kmitočtová charakteristika zesilovače s rozprostřeným zesílením



7.1.2. Citlivosti kmitočtové charakteristiky zesilovače s rozprostřeným zesílením

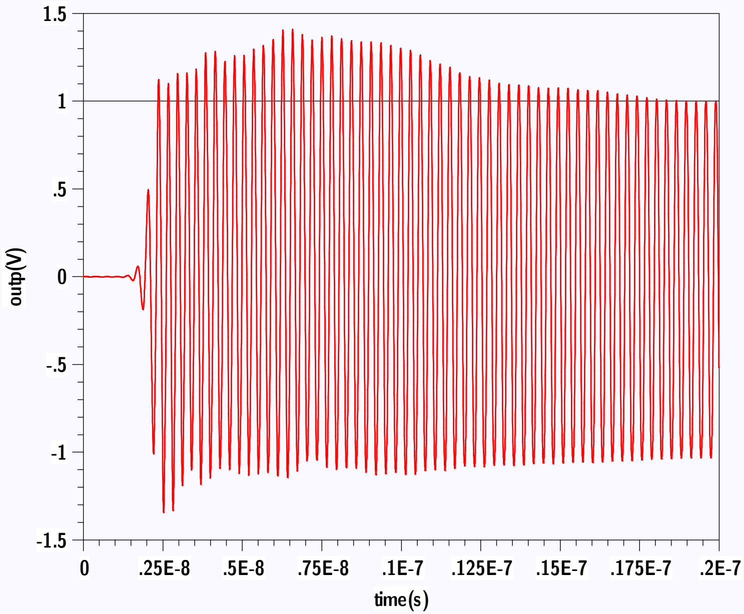


7.2. Oscilátor s rozprostřeným zesílením

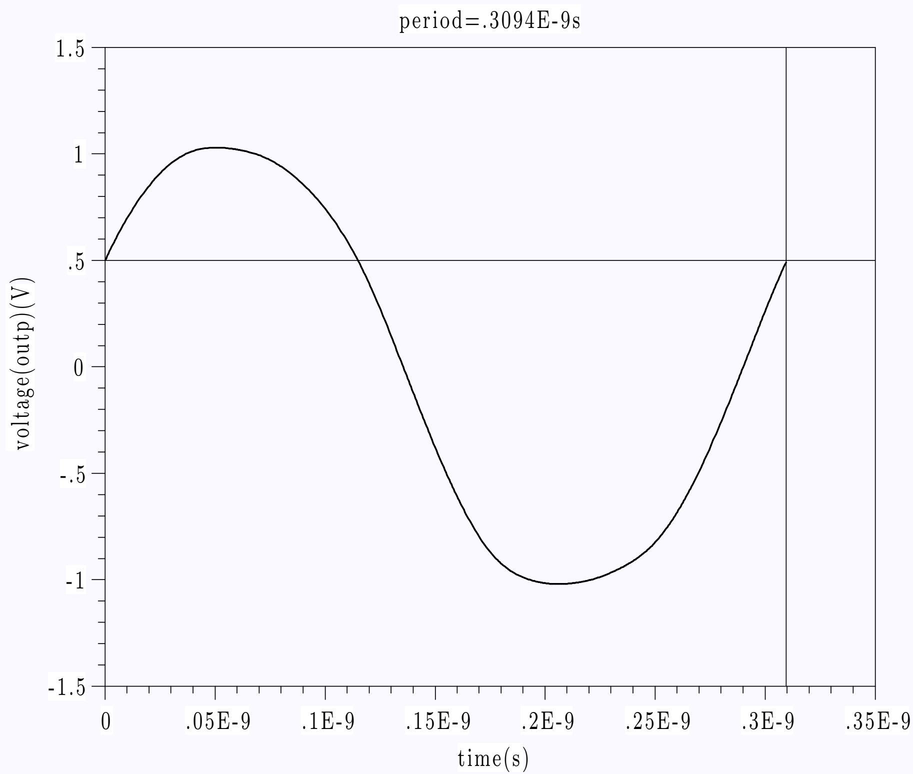


Domovská stránka
Strana 21 z 30
Zpět
Celá obrazovka
Zavřít
Konec

7.2.1. Velmi komplikovaný (a dlouhý) přechodný děj zesilovače s rozprostřeným zesílením



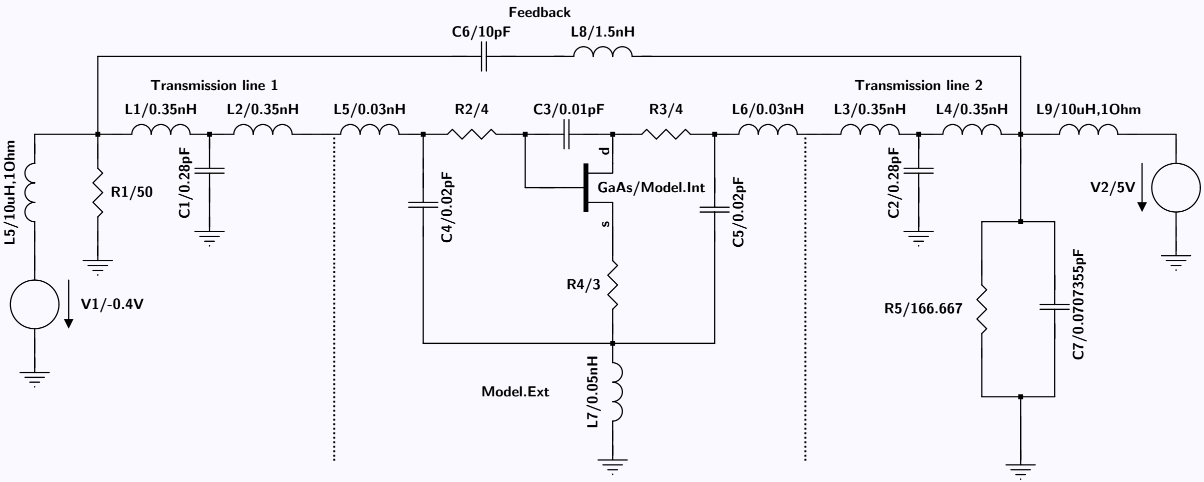
7.2.2. Ustálená periodická odezva detekovaná průchody úrovní V_{outp} = 0.5 V:



7.2.3. Porovnání klasické numerické integrace s aplikací ϵ-algoritmu

Použitá metoda	Počty integračních kroků						
	1.	2.	3.	4.	5.	6.	Celkem
1. iterace	10	42	166	512	1893	9931	12554
2. iterace	3	2	24	419	2258	11454	14160
3. iterace	3	2	21	463	2467	11256	14212
4. iterace	3	2	28	430	2078	8827	11368
Extrapol.	19	48	239	1824	8696	41468	**52294**
Klasická	10	38	254	3631	20172	94829	**118934**

8. Zpětnovazební mikrovlnný oscilátor



Utilized method	Numbers of integration steps (for orders and total)						
	1st	2nd	3rd	4th	5th	6th	Total
1st iter.	25	285	877	1023	778	723	3711
2nd iter.	4	84	318	723	1611	3179	5919
Extrapol.	29	369	1195	1746	2389	3902	**9630**
Classical	29	424	1573	3047	7558	16183	**28814**

9. Oscilátor s tunelovou diodou

Tunelová dioda je reprezentována polynomem jedenáctého řádu

```
"Tunnel diode oscillator
e 2 0 bias(x,t)
l 2 1 2.5uH
c 1 0 100pF
f1^1 0 tunnel
(
u1,u0,0,
?p1 = 1-1( 0  :1  )
?p2 =-1  (-1+1:0  )
?p3 = 1+1( 0  :1+2)
?p4 =-1+2(-1+3:0  )
?p5 = 1+3( 0  :1+4)
?p6 =-1+4(-1+5:0  )
?p7 = 1+5( 0  :1+6)
?p8 =-1+5(-1+6:0  )
?p9 = 1+5( 0  :1+6)
?p10=-1+5(-1+6:0  )
?p11= 1+5( 0  :1+6)
)
*
```

definovaného funkcí (polynom efektivně naprogramovaný Hornerovým schématem):

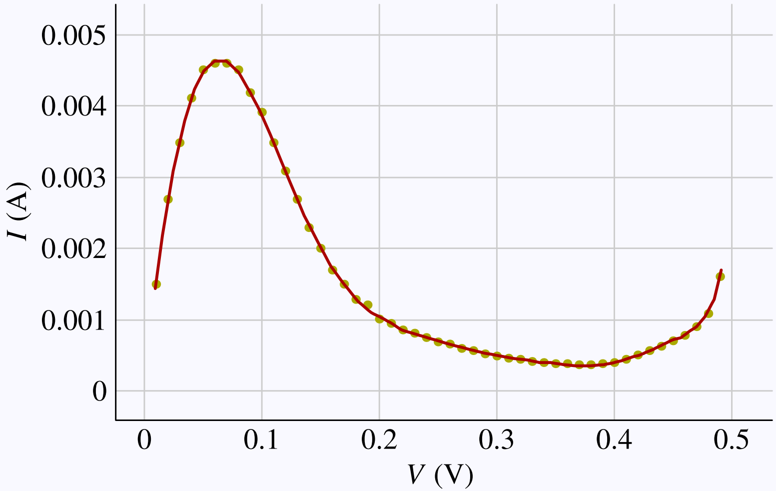
```
      function tunnel(uplus,uminus,
     -                p0,p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11)
      implicit double precision(a-h,o-z)
      u=uplus-uminus
      q11   =u*(p11    )
      q10   =u*(p10+q11)
      q9    =u*(p9 +q10)
      q8    =u*(p8 +q9 )
      q7    =u*(p7 +q8 )
      q6    =u*(p6 +q7 )
      q5    =u*(p5 +q6 )
      q4    =u*(p4 +q5 )
      q3    =u*(p3 +q4 )
      q2    =u*(p2 +q3 )
      q1    =u*(p1 +q2 )
      tunnel=  (p0 +q1 )
      end
```

Optimalizace nalezla po (pouhých) sedmi iteracích Levenbergova algoritmu následující hodnoty koeficientů polynomu ("grafické" pole ukazuje polohu nalezeného parametru mezi povoleným minimem a maximem):

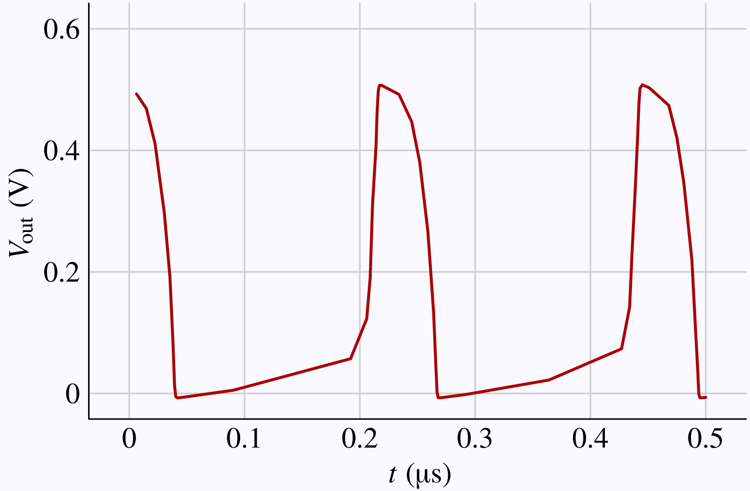
```
P1      .177     [    .   A.    .    .    .    .    .    .    .    ]
P2     -2.72     [    .    .    .    .    .    .    .B   .    .    ]
P3      36.8     [    .    .    .  C .    .    .    .    .    .    ]
P4     -563      [    .    .    .    . D  .    .    .    .    .    ]
P5      5800     [    .    .    .    .    .   E.    .    .    .    ]
P6     -36300    [    .    .    .    .    .    . F  .    .    .    ]
P7      142000   [    . G  .    .    .    .    .    .    .    .    ]
P8     -351000   [    .    .    .    .    .    . H  .    .    .    ]
P9      534000   [    .    .    .    .    . I  .    .    .    .    ]
P10    -459000   [    .    .    .    .    . J  .    .    .    .    ]
P11     170000   [    .   K.    .    .    .    .    .    .    .    ]
```


Domovská stránka
◀◀ ▶▶
◀ ▶
Strana 28 z 30
Zpět
Celá obrazovka
Zavřít
Konec

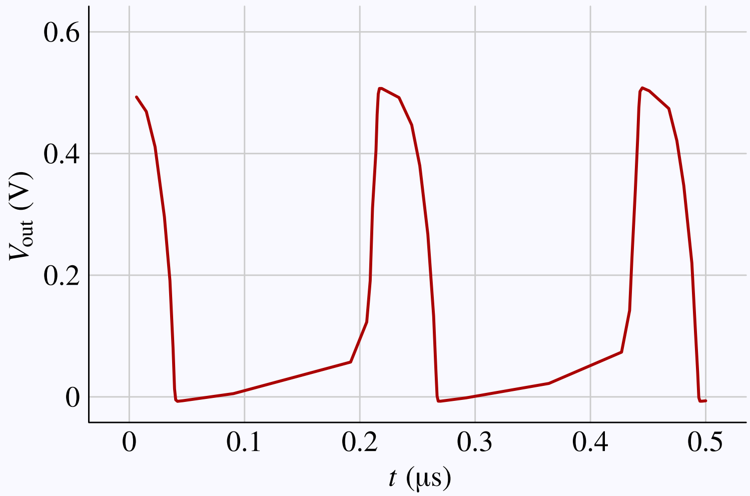
Interpolace naměřených bodů identifikovaným polynomem je velmi zdařilá:



Časová analýza prokazuje pravidelné (a netlumené!) kmity oscilátoru:

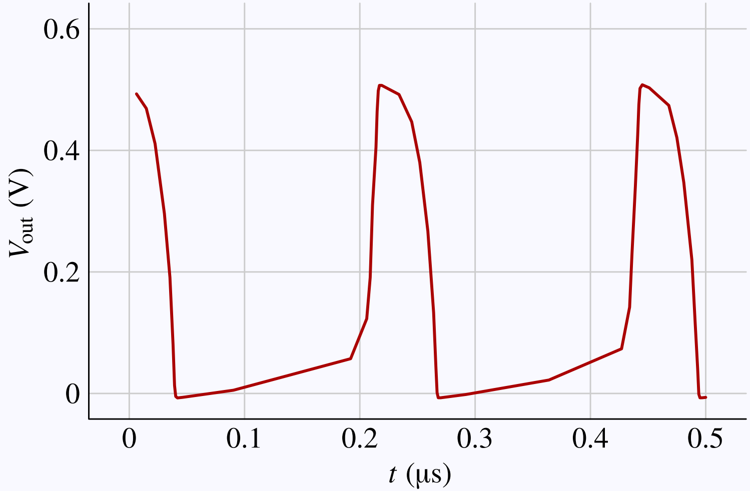


Časová analýza prokazuje pravidelné (a netlumené!) kmity oscilátoru:



Je zjevné, že kmity jsou velmi neharmonické (což často v učebnicích není zmiňováno) a oscilátor tedy (pro použití v radiotechnice) musí být doprovázen filtrem.

Časová analýza prokazuje pravidelné (a netlumené!) kmity oscilátoru:



Je zjevné, že kmity jsou velmi neharmonické (což často v učebnicích není zmiňováno) a oscilátor tedy (pro použití v radiotechnice) musí být doprovázen filtrem.

Tunelový jev ovšem patří k "nejtišším" v přírodě a oscilátory tohoto typu se tedy vyznačují velmi malým šumem.