Hledání kořenů rovnic jedné reálné proměnné – metoda půlení intervalů –

Michal Čihák

23. října 2012

Problém hledání kořenů rovnice $f(x) = 0$

- jeden ze základních problémů numerické matematiky
- zároveň i jeden z nejstarších problémů
- například Babylonské hliněné destičky z období 1700 před n. l. uvádějí přibližnou hodnotu jednoho z kořenů rovnice $x^2 - 2 = 0$ v šedesátkové soustavě
- v dnešní zápisu po převedení do desítkové soustavy se jedná o hodnotu $1{,}414\,222$
- shoda se skutečnou hodnotou čísla $\sqrt{2}$ je na desetitisíciny
- výzkum v této oblasti pokračuje i dnes

Problém hledání kořenů rovnice $f(x) = 0$

- jeden ze základních problémů numerické matematiky
- zároveň i jeden z nejstarších problémů
- například Babylonské hliněné destičky z období 1700 před n. l. uvádějí přibližnou hodnotu jednoho z kořenů rovnice $x^2 - 2 = 0$ v šedesátkové soustavě
- v dnešní zápisu po převedení do desítkové soustavy se jedná o hodnotu $1{,}414\,222$
- shoda se skutečnou hodnotou čísla $\sqrt{2}$ je na desetitisíciny
- výzkum v této oblasti pokračuje i dnes

Problém hledání kořenů rovnice $f(x) = 0$

- jeden ze základních problémů numerické matematiky
- zároveň i jeden z nejstarších problémů
- například Babylonské hliněné destičky z období 1700 před n. l. uvádějí přibližnou hodnotu jednoho z kořenů rovnice $x^2 - 2 = 0$ v šedesátkové soustavě
- v dnešní zápisu po převedení do desítkové soustavy se jedná o hodnotu $1{,}414\,222$
- shoda se skutečnou hodnotou čísla $\sqrt{2}$ je na desetitisíciny
- výzkum v této oblasti pokračuje i dnes

Problém hledání kořenů rovnice $f(x) = 0$

- jeden ze základních problémů numerické matematiky
- zároveň i jeden z nejstarších problémů
- například Babylonské hliněné destičky z období 1700 před n. l. uvádějí přibližnou hodnotu jednoho z kořenů rovnice $x^2 - 2 = 0$ v šedesátkové soustavě
- v dnešní zápisu po převedení do desítkové soustavy se jedná o hodnotu $1{,}414\,222$
- shoda se skutečnou hodnotou čísla $\sqrt{2}$ je na desetitisíciny
- výzkum v této oblasti pokračuje i dnes

Problém hledání kořenů rovnice $f(x) = 0$

- jeden ze základních problémů numerické matematiky
- zároveň i jeden z nejstarších problémů
- například Babylonské hliněné destičky z období 1700 před n. l. uvádějí přibližnou hodnotu jednoho z kořenů rovnice $x^2 - 2 = 0$ v šedesátkové soustavě
- v dnešní zápisu po převedení do desítkové soustavy se jedná o hodnotu $1{,}414\,222$
- shoda se skutečnou hodnotou čísla $\sqrt{2}$ je na desetitisíciny
- výzkum v této oblasti pokračuje i dnes

Problém hledání kořenů rovnice $f(x) = 0$

- jeden ze základních problémů numerické matematiky
- zároveň i jeden z nejstarších problémů
- například Babylonské hliněné destičky z období 1700 před n. l. uvádějí přibližnou hodnotu jednoho z kořenů rovnice $x^2 - 2 = 0$ v šedesátkové soustavě
- v dnešní zápisu po převedení do desítkové soustavy se jedná o hodnotu $1{,}414\,222$
- shoda se skutečnou hodnotou čísla $\sqrt{2}$ je na desetitisíciny
- výzkum v této oblasti pokračuje i dnes

Metoda půlení intervalů

- nejjednodušší metoda
- s její pomocí můžeme nalézt řešení rovnice $f(x) = 0$ na intervalu $\langle a, b \rangle$ v podstatě s libovolnou přesností (kterou umožňuje náš počítač)
- předpokladem ale je, že funkce $f(x)$ je na intervalu $\langle a, b \rangle$ spojitá a $f(a)$ a $f(b)$ mají rozdílná znaménka
- pro jednoduchost budeme předpokládat, že rovnice má na intervalu $\langle a, b \rangle$ jeden kořen
- tato metoda však funguje i pro případ, že na intervalu $\langle a, b \rangle$ má rovnice více než jeden kořen

Metoda půlení intervalů

- nejjednodušší metoda
- s její pomocí můžeme nalézt řešení rovnice $f(x) = 0$ na intervalu $\langle a, b \rangle$ v podstatě s libovolnou přesností (kterou umožňuje náš počítač)
- předpokladem ale je, že funkce $f(x)$ je na intervalu $\langle a, b \rangle$ spojitá a $f(a)$ a $f(b)$ mají rozdílná znaménka
- pro jednoduchost budeme předpokládat, že rovnice má na intervalu $\langle a, b \rangle$ jeden kořen
- tato metoda však funguje i pro případ, že na intervalu $\langle a, b \rangle$ má rovnice více než jeden kořen

Metoda půlení intervalů

- nejjednodušší metoda
- s její pomocí můžeme nalézt řešení rovnice $f(x) = 0$ na intervalu $\langle a, b \rangle$ v podstatě s libovolnou přesností (kterou umožňuje náš počítač)
- předpokladem ale je, že funkce $f(x)$ je na intervalu $\langle a, b \rangle$ spojitá a $f(a)$ a $f(b)$ mají rozdílná znaménka
- pro jednoduchost budeme předpokládat, že rovnice má na intervalu $\langle a, b \rangle$ jeden kořen
- tato metoda však funguje i pro případ, že na intervalu $\langle a, b \rangle$ má rovnice více než jeden kořen

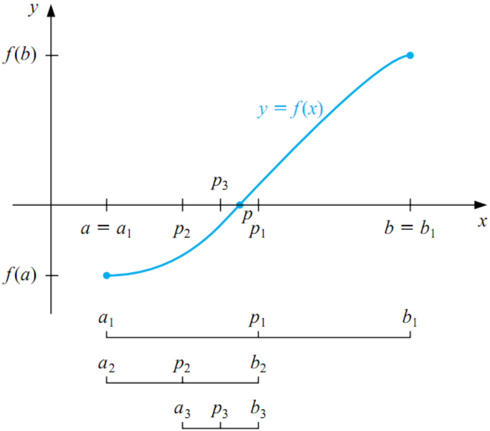
Metoda půlení intervalů

- nejjednodušší metoda
- s její pomocí můžeme nalézt řešení rovnice $f(x) = 0$ na intervalu $\langle a, b\rangle$ v podstatě s libovolnou přesností (kterou umožňuje náš počítač)
- předpokladem ale je, že funkce $f(x)$ je na intervalu $\langle a, b\rangle$ spojitá a $f(a)$ a $f(b)$ mají rozdílná znaménka
- pro jednoduchost budeme předpokládat, že rovnice má na intervalu $\langle a, b\rangle$ jeden kořen
- tato metoda však funguje i pro případ, že na intervalu $\langle a, b\rangle$ má rovnice více než jeden kořen

Metoda půlení intervalů

- nejjednodušší metoda
- s její pomocí můžeme nalézt řešení rovnice $f(x) = 0$ na intervalu $\langle a, b\rangle$ v podstatě s libovolnou přesností (kterou umožňuje náš počítač)
- předpokladem ale je, že funkce $f(x)$ je na intervalu $\langle a, b\rangle$ spojitá a $f(a)$ a $f(b)$ mají rozdílná znaménka
- pro jednoduchost budeme předpokládat, že rovnice má na intervalu $\langle a, b\rangle$ jeden kořen
- tato metoda však funguje i pro případ, že na intervalu $\langle a, b\rangle$ má rovnice více než jeden kořen

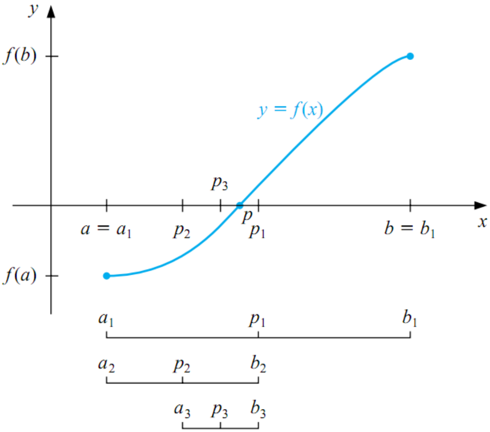
Algoritmus metody půlení intervalů



na začátku položíme $a_1 = a, b_1 = b$ a najdeme střed intervalu $\langle a_1, b_1 \rangle$ pomocí vztahu

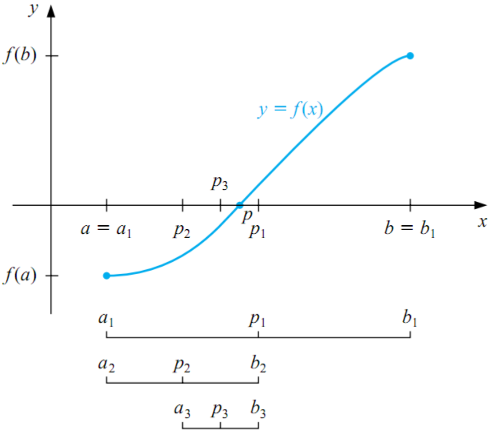
$$p_1 = a_1 + \frac{b_1 - a_1}{2}$$

Algoritmus metody půlení intervalů



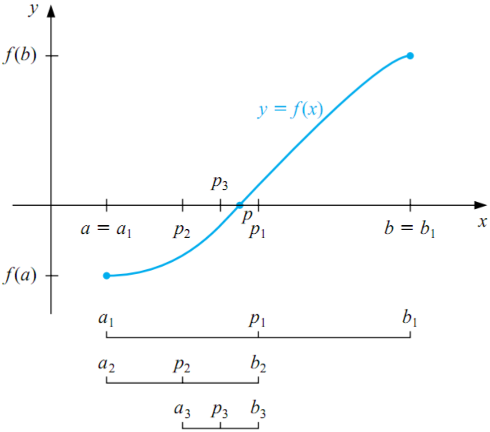
pokud je $f(p_1) = 0$, pak je p_1 hledaným kořenem rovnice

Algoritmus metody půlení intervalů



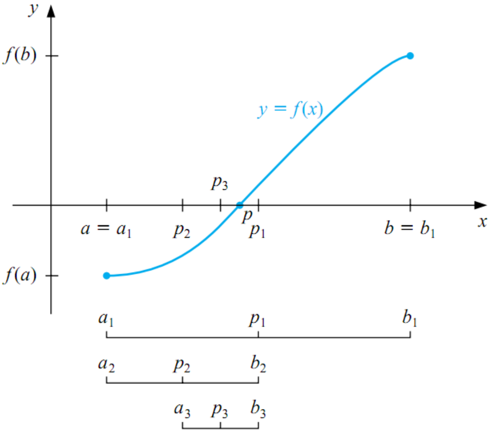
v opačném případě má $f(p_1)$ stejné znaménko buď jako $f(a_1)$, nebo jako $f(b_1)$

Algoritmus metody půlení intervalů



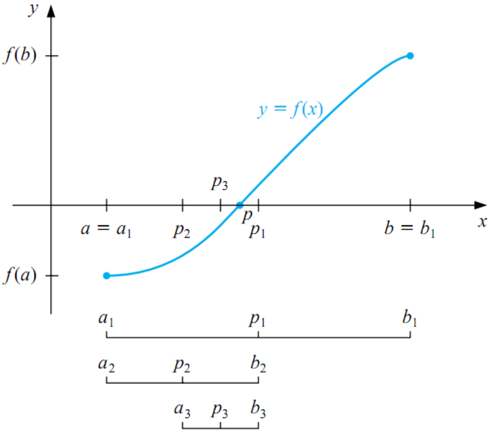
pokud má $f(p_1)$ stejné znaménko jako $f(a_1)$ stejná znaménka, pak hledaný kořen rovnice leží v intervalu $\langle p_1, b_1 \rangle$ a položíme $a_2 = p_1, b_2 = b_1$

Algoritmus metody půlení intervalů



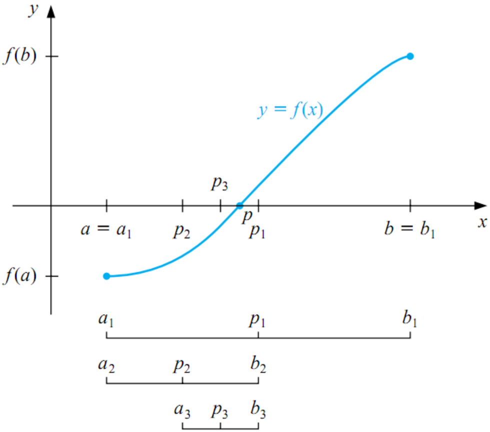
pokud má $f(p_1)$ stejné znaménko jako $f(b_1)$, pak hledaný kořen rovnice leží v intervalu $\langle a_1, p_1 \rangle$ a položíme $a_2 = a_1, b_2 = p_1$

Algoritmus metody půlení intervalů



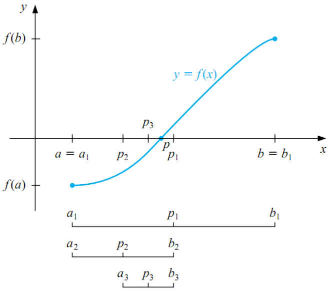
nyní opakujeme stejný proces na interval $\langle a_2, b_2 \rangle$, poté na interval $\langle a_3, b_3 \rangle$, atd.

Algoritmus metody půlení intervalů



každý nově vzniklý interval obsahuje hledaný kořen a jeho délka je poloviční oproti předchozímu intervalu – odtud název metody

Algoritmus metody půlení intervalů – shrnutí



Interval $\langle a_{n+1}, b_{n+1}\rangle$, kde $n > 0$, obsahující kořen rovnice $f(x) = 0$ získáme z intervalu $\langle a_n, b_n\rangle$ tak, že nejprve určíme střed intervalu $\langle a_n, b_n\rangle$ pomocí vztahu

$$p_n = a_n + \frac{b_n - a_n}{2}.$$

Poté položíme $a_{n+1} = a_n$ a $b_{n+1} = p_n$, pokud je $f(a_n) \cdot f(p_n) < 0$, nebo $a_{n+1} = p_n$ a $b_{n+1} = b_n$, pokud je $f(a_n) \cdot f(p_n) > 0$.

Metoda půlení intervalů

Existují 3 základní kritéria pro ukončení algoritmu metody půlení intervalu:

1. některý střed intervalu je přímo kořenem rovnice
2. délka intervalu $\langle a_n, b_n \rangle$ klesne pod nějakou předem danou toleranci – označme ji TOL
3. počet iterací algoritmu překročí předem danou mez N_0

Metoda půlení intervalů

Existují 3 základní kritéria pro ukončení algoritmu metody půlení intervalu:

1. některý střed intervalu je přímo kořenem rovnice
2. délka intervalu $\langle a_n, b_n \rangle$ klesne pod nějakou předem danou toleranci – označme ji TOL
3. počet iterací algoritmu překročí předem danou mez N_0

Metoda půlení intervalů

Existují 3 základní kritéria pro ukončení algoritmu metody půlení intervalu:

1. některý střed intervalu je přímo kořenem rovnice
2. délka intervalu $\langle a_n, b_n \rangle$ klesne pod nějakou předem danou toleranci – označme ji TOL
3. počet iterací algoritmu překročí předem danou mez N_0

Odhad počtu iterací

Pro zahájení metody půlení intervalů musíme nejprve nalézt interval $\langle a, b\rangle$ tak, aby platilo

$$f(a) \cdot f(b) < 0.$$

Odhad počtu iterací

Pro zahájení metody půlení intervalů musíme nejprve nalézt interval $\langle a, b\rangle$ tak, aby platilo

$$f(a) \cdot f(b) < 0.$$

Pro vzdálenost středu p_1 intervalu $\langle a, b\rangle$ od kořene p ležícího v tomto intervalu platí

$$|p_1 - p| \leq \frac{b - a}{2}.$$

Odhad počtu iterací

Pro zahájení metody půlení intervalů musíme nejprve nalézt interval $\langle a, b \rangle$ tak, aby platilo

$$f(a) \cdot f(b) < 0.$$

Pro vzdálenost středu p_1 intervalu $\langle a, b \rangle$ od kořene p ležícího v tomto intervalu platí

$$|p_1 - p| \leq \frac{b-a}{2}.$$

Každá následující iterace zmenší délku uvažovaného intervalu na polovinu a tedy obecně platí

$$|p_n - p| \leq \frac{b-a}{2^n}.$$

Odhad počtu iterací

Pro zahájení metody půlení intervalů musíme nejprve nalézt interval $\langle a, b\rangle$ tak, aby platilo

$$f(a) \cdot f(b) < 0.$$

Pro vzdálenost středu p_1 intervalu $\langle a, b\rangle$ od kořene p ležícího v tomto intervalu platí

$$|p_1 - p| \leq \frac{b-a}{2}.$$

Každá následující iterace zmenší délku uvažovaného intervalu na polovinu a tedy obecně platí

$$|p_n - p| \leq \frac{b-a}{2^n}.$$

Pokud zajistíme, aby

$$\frac{b-a}{2^n} < TOL,$$

potom máme zaručeno, že absolutní chyba aproximace kořene nepřekročí předem danou hodnotu TOL.

Odhad počtu iterací

Pro zahájení metody půlení intervalů musíme nejprve nalézt interval $\langle a, b\rangle$ tak, aby platilo

$$f(a) \cdot f(b) < 0.$$

Pro vzdálenost středu p_1 intervalu $\langle a, b\rangle$ od kořene p ležícího v tomto intervalu platí

$$|p_1 - p| \leq \frac{b-a}{2}.$$

Každá následující iterace zmenší délku uvažovaného intervalu na polovinu a tedy obecně platí

$$|p_n - p| \leq \frac{b-a}{2^n}.$$

Pokud zajistíme, aby

$$\frac{b-a}{2^n} < TOL,$$

potom máme zaručeno, že absolutní chyba aproximace kořene nepřekročí předem danou hodnotu TOL. Z předchozí nerovnice můžeme určit počet iterací potřebný k dosažení předem dané přesnosti

$$\frac{b-a}{TOL} < 2^n \qquad \Rightarrow \qquad n > \log_2\left(\frac{b-a}{TOL}\right).$$

Volba výchozího intervalu

- Protože odhad počtu iterací závisí na délce výchozího intervalu $\langle a, b \rangle$, snažíme se jej volit co nejkratší.
- Například pro $f(x) = 2x^3 - x^2 + x - 1 = 0$ můžeme použít výchozí interval $\langle -4, 4 \rangle$, protože $f(-4) \cdot f(4) < 0$, nebo výchozí interval $\langle 0, 1 \rangle$, protože $f(0) \cdot f(1) < 0$. Pokud použijeme druhý z intervalů namísto prvního, počet iterací se třikrát sníží.

Volba výchozího intervalu

- Protože odhad počtu iterací závisí na délce výchozího intervalu $\langle a, b\rangle$, snažíme se jej volit co nejkratší.
- Například pro $f(x) = 2x^3 - x^2 + x - 1 = 0$ můžeme použít výchozí interval $\langle -4, 4\rangle$, protože $f(-4) \cdot f(4) < 0$, nebo výchozí interval $\langle 0, 1\rangle$, protože $f(0) \cdot f(1) < 0$. Pokud použijeme druhý z intervalů namísto prvního, počet iterací se třikrát sníží.

Příklad

Zadání: Najděte kořen rovnice $x^3 + 4x^2 - 10 = 0$ v intervalu $\langle 1, 2 \rangle$ s tolerancí $0{,}0005$.

Příklad

Zadání: Najděte kořen rovnice $x^3 + 4x^2 - 10 = 0$ v intervalu $\langle 1, 2 \rangle$ s tolerancí $0{,}0005$.

Řešení: Nejprve určíme počet iterací potřebný k dosažení předem dané přesnosti

$$n > \log_2 \left(\frac{2-1}{0{,}0005} \right) \doteq 10{,}96.$$

Příklad

Zadání: Najděte kořen rovnice $x^3 + 4x^2 - 10 = 0$ v intervalu $\langle 1, 2 \rangle$ s tolerancí $0{,}0005$.

Řešení: Nejprve určíme počet iterací potřebný k dosažení předem dané přesnosti

$$n > \log_2 \left(\frac{2-1}{0{,}0005} \right) \doteq 10{,}96.$$

Měli bychom tedy provést alespoň 11 iterací algoritmu půlení intervalu. Výsledky jsou shrnuty v tabulce:

n	a_n	b_n	p_n	$f(p_n)$
1	1.0000000000	2.0000000000	1.5000000000	2.3750000000
2	1.0000000000	1.5000000000	1.2500000000	−1.7968750000
3	1.2500000000	1.5000000000	1.3750000000	0.1621093750
4	1.2500000000	1.3750000000	1.3125000000	−0.8483886719
5	1.3125000000	1.3750000000	1.3437500000	−0.3509826660
6	1.3437500000	1.3750000000	1.3593750000	−0.0964088440
7	1.3593750000	1.3750000000	1.3671875000	0.0323557854
8	1.3593750000	1.3671875000	1.3632812500	−0.0321499705
9	1.3632812500	1.3671875000	1.3652343750	0.0000720248
10	1.3632812500	1.3652343750	1.3642578125	−0.0160466908
11	1.3642578125	1.3652343750	1.3647460938	−0.0079892628

Výhody metody půlení intervalů

- jednoduchý princip a snadná implementace (naprogramování algoritmu v konkrétním programovacím jazyce)
- jsou-li dodrženy předpoklady metody, pak metoda vždy konverguje k některému z kořenů v daném intervalu
- je k dispozici jednoduché kritérium pro stanovení počtu iterací potřebného pro dosažení dané přesnosti

Výhody metody půlení intervalů

- jednoduchý princip a snadná implementace (naprogramování algoritmu v konkrétním programovacím jazyce)
- jsou-li dodrženy předpoklady metody, pak metoda vždy konverguje k některému z kořenů v daném intervalu
- je k dispozici jednoduché kritérium pro stanovení počtu iterací potřebného pro dosažení dané přesnosti

Výhody metody půlení intervalů

- jednoduchý princip a snadná implementace (naprogramování algoritmu v konkrétním programovacím jazyce)
- jsou-li dodrženy předpoklady metody, pak metoda vždy konverguje k některému z kořenů v daném intervalu
- je k dispozici jednoduché kritérium pro stanovení počtu iterací potřebného pro dosažení dané přesnosti

Nevýhody metody půlení intervalů

- oproti jiným metodám je konvergence pomalá
- ačkoliv se v průběhu algoritmu k přesné hodnotě kořenu někdy velmi přiblížíme, v následujícím kroku se můžeme zase od kořenu vzdálit - viz hodnoty p_9 a p_{10} v tabulce u předchozího příkladu

Nevýhody metody půlení intervalů

- oproti jiným metodám je konvergence pomalá
- ačkoliv se v průběhu algoritmu k přesné hodnotě kořenu někdy velmi přiblížíme, v následujícím kroku se můžeme zase od kořenu vzdálit - viz hodnoty p_9 a p_{10} v tabulce u předchozího příkladu

Rizika implementace metody na počítači

- při výpočtech musíme vzít v úvahu riziko ztráty přesnosti v důsledku zaokrouhlovacích chyb – proto například používáme vztah $p_n = a_n + \frac{b_n - a_n}{2}$ namísto matematicky ekvivalentního vztahu $p_n = \frac{a_n + b_n}{2}$
- při výpočtu součinu v kritériu $f(a_n) \cdot f(p_n) < 0$ by se mohlo stát, že absolutní hodnota výsledku bude menší než nejmenší kladné číslo, které lze uložit v daném číselném formátu v počítači (nebo naopak tato hodnota překročí maximální hodnotu daného číselného formátu) – proto raději používáme kritérium $\mathrm{sgn}(f(a_n)) \cdot \mathrm{sgn}(f(p_n)) < 0$, kde sgn je tzv. znaménková funkce.

Rizika implementace metody na počítači

- při výpočtech musíme vzít v úvahu riziko ztráty přesnosti v důsledku zaokrouhlovacích chyb – proto například používáme vztah $p_n = a_n + \frac{b_n - a_n}{2}$ namísto matematicky ekvivalentního vztahu $p_n = \frac{a_n + b_n}{2}$
- při výpočtu součinu v kritériu $f(a_n) \cdot f(p_n) < 0$ by se mohlo stát, že absolutní hodnota výsledku bude menší než nejmenší kladné číslo, které lze uložit v daném číselném formátu v počítači (nebo naopak tato hodnota překročí maximální hodnotu daného číselného formátu) – proto raději používáme kritérium $\mathrm{sgn}(f(a_n)) \cdot \mathrm{sgn}(f(p_n)) < 0$, kde sgn je tzv. znaménková funkce.