Soustavy nelineárních rovnic pomocí systému Maple. Newtonova metoda.

Úvod

Mnoho technických problémů vede na řešení matematických úloh, které se následně převedou na úlohy řešení soustav nelineárních rovnic

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}. \tag{1}$$

Rovnice (1) znamená soustavu n rovnic zapsanou ve vektorovém tvaru, tj. $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ je vektorová funkce a soustava se řeší pro neznámou n-tici hodnot $\mathbf{x}$. Řešení rovnice (1) pro případ $n = 1$ je popsán v textu **Řešení rovnic**(Matematika I), na který se zde navazuje. Použití Newtonovy metody pro případ $n = 1$ nalezneme v příkladech pro zmíněný text.

Pro řešení soustavy nelineárních rovnic používáme velmi často numerické iterační metody. Zvláště často se používá Newtonova metoda, která je zobecněním Newtonovy iterační metody používané v případě jedné rovnice. Jak níže ukážeme, jedná se sice o analogickou metodu, ve skutečnosti jsou však mezi jedno- a vícedimenzionálním případem podstatné rozdíly. V případě jedné rovnice, tj. $n = 1$, je možné snadno získat dobré informace o poloze kořene. Pokud polohu blíže neznáme, lze užít k získání řešení některou z metod, která vždy konverguje, například metodu půlení intervalu. V případě více dimenzí se informace o poloze řešení získávají nesrovnatelně obtížněji, než v případě $n = 1$. Navíc nejsou známy metody, které vždy konvergují při řešení (1) pro $n \geq 2$.

V tomto textu budeme dále uvažovat pouze případ $n = 2$, tj. **soustavu dvou nelineárních rovnic**

$$\begin{aligned} f_1(x,y) &= 0 \\ f_2(x,y) &= 0\ , \end{aligned} \tag{2}$$

kde f_1, f_2 jsou reálné funkce dvou reálných proměnných x, y.

V druhé části ukážeme, že i pro $n = 2$ je obtížné nejen určit počet řešení úlohy, ale také tyto kořeny separovat do oblastí tak, aby vhodné zvolení počáteční aproximace řešení vedlo v iterační metodě k nalezení hledaného řešení soustavy.

V třetí části tohoto textu se budeme zabývat Newtonovou iterační metodou pro řešení soustav nelineárních rovnic. Popíšeme algoritmus této metody a ukážeme, jak lze Newtonovu metodu naprogramovat v systému Maple. Poznamenejme, že existují i jiné metody pro řešení soustav typu (1); některé z nich jsou modifikací zde probírané Newtonovy metody.

Řešitelnost dvou nelineárních rovnic

Jak obtížné je nalézt přibližnou polohu řešení soustavy nelineárních rovnic lze nejlépe demonstrovat již pro případ $n = 2$. Pro $n > 2$ by diskuse řešitelnosti soustav byla daleko obtížnější.

V případě dvou nelineárních rovnic (2) si lze udělat následující geometrickou představu. Předpokládejme, že soustava (2) má jenom izolované kořeny. Funkce f_1 a f_2 jsou funkce dvou proměnných, jejichž nulové vrstevnice oddělují v $\mathbb{R}^2$ oblasti, kde jsou tyto funkce kladné a kde záporné. Právě body, kde se nulová vrstevnice funkce f_1 protíná s nulovou vrstevnicí funkce f_2, jsou řešením soustavy (2).

V příkladech 1 - 3 znázorníme nulové vrstevnice funkcí f_1 a f_2 pomocí systému Maple příkazem `implicitplot` z balíku `plots`. Přáli bychom si, obdobně jako v případě $n = 1$, abychom našli

takové oblasti v $\mathbb{R}^2$ a v nich první přiblížení, které by zaručilo konvergenci numerické metody k řešení. Nahlédneme, že separovat průsečíky nulových vrstevnic do takových oblastí není pro $n = 2$ jednoduché.

Příklad 1. Definujme soustavu nelineárních rovnic

```
> f1:=x-y^2+1;
```

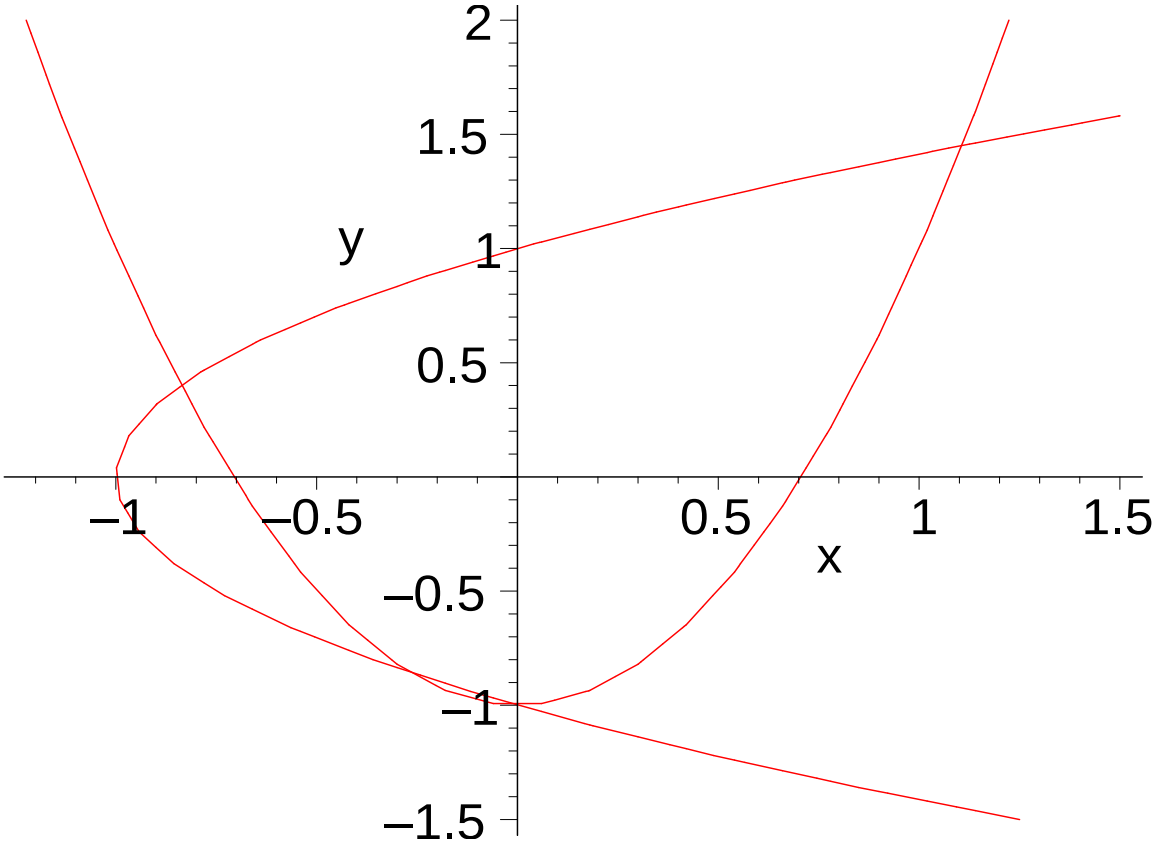
$$f1 := x - y^2 + 1$$

```
> f2:=2*x^2-y-1;
```

$$f2 := 2\,x^2 - y - 1$$

a nakresleme grafy příslušných nulových vrstevnic.

```
> plots[implicitplot]({f1=0,f2=0},x=-1.5..1.5,y=-1.5..2);
```



Tato soustava má čtyři řešení, z nichž dvě jsou relativně blízko sebe, jedno je zřejmě $\mathbf{x}_1 = [0; -1]$.

Pro řešení jedné algebraické rovnice jsme v Maple používali příkaz `fsolve` (zadání ve tvaru `fsolve(eqns, vars, options)`). Pro soustavy dvou rovnic lze použít stejný příkaz, kde je první argument dvojice rovnic (bez nulových pravých stran). Parametr `vars` nemusíme zadávat, tj. ani proměnné, ani počáteční přiblížení. Následujícím příkazem dostáváme jeden z kořenů soustavy, tedy $\mathbf{x}_1$:

```
> reseni1:=fsolve({f1,f2});
```

$$reseni1 := \{x = -0.6934800000\,10^{-24},\ y = -1.000000000\}$$

Další řešení se pokusíme nalézt tak, že zadáme počáteční přiblížení, které se zdá být dostatečně blízké jinému řešení. Dostáváme ale znovu první nalezené řešení:

```
> reseni1:=fsolve({f1,f2},{x=0,y=-0.7});
```

$$reseni1 := \{x = 0.,\ y = -1.000000000\}$$

Již při drobné změně počátečního přiblížení dostaneme druhé hledané řešení $\mathbf{x}_2$, které je blízké řešení prvnímu. Dále pak lze systémem Maple (máme-li štěstí při volbě počátečních přiblížení) spočítat ostatní kořeny soustavy $\mathbf{x}_3$, $\mathbf{x}_4$.

```
> reseni2:=fsolve({f1,f2},{x=0,y=-0.6});
```

$$reseni2 := \{x = -0.2695944364,\ y = -0.8546376797\}$$

```
> reseni3:=fsolve({f1,f2},{x=-1,y=1});
```

$$reseni3 := \{x = -0.8375654353,\ y = 0.4030317168\}$$

```
> reseni4:=fsolve({f1,f2},{x=1,y=1});
```

$$reseni4 := \{y = 1.451605963,\ x = 1.107159872\}$$

Dalším příkladem je soustava rovnic, kterou rovněž znázorníme graficky. Stejně budeme hledat průsečíky dvou parabol, jako v příkladu 1.

Příklad 2. Soustava

```
> f1:=x-y^2;
```

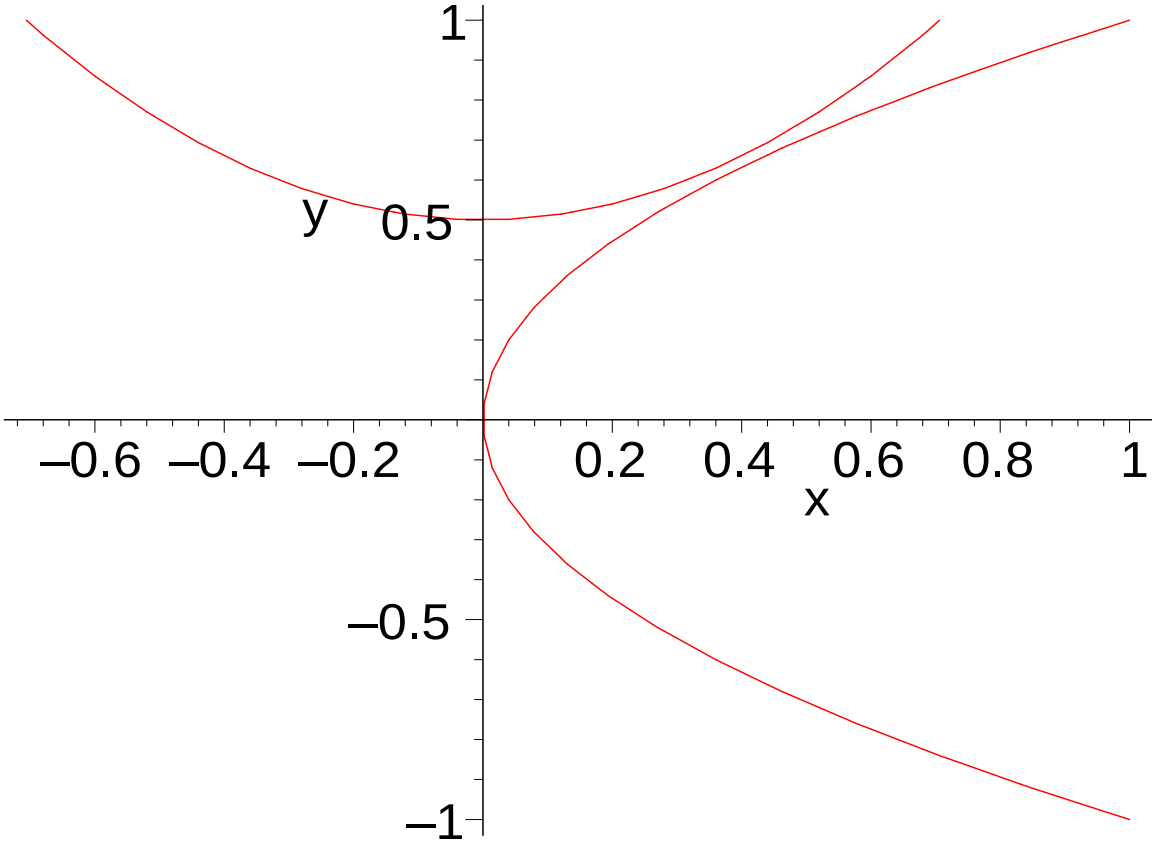
$$f1 := x - y^2$$

```
> f2:=2*x^2-2*y+1;
```

$$f2 := 2\,x^2 - 2\,y + 1$$

nemá řešení. Grafy nulových vrstevnic se neprotínají.

```
> plots[implicitplot]({f1=0,f2=0},x=-1..1,y=-1..1);
```



Následujícím příkazem skutečně nedostaneme řešení soustavy. Systém Maple pouze přepíše zadání

do jiného tvaru:

```
> reseni1:=fsolve({f1,f2});
```

$$reseni1 := \text{fsolve}(\{2\,x^2 - 2\,y + 1,\ x - y^2\},\ \{x,\ y\})$$

Poznamenejme, že systém Maple by správně nenalezl řešení ani kdybychom do příkazu zadali co nejbližší počáteční přiblížení k očekávanému řešení.
V některých případech může být hranice mezi žádným řešením soustavy a právě jedním řešením zprvu nepostřehnutelná. V těchto případech je nutné provést systémem Maple detailnější graf.

V dalším příkladu má soustava právě jedno řešení, ale systém Maple při hledání tohoto řešení selhává, i když volíme počáteční přiblížení dostatečně blízko hledanému řešení.

Příklad 3. Soustava

```
> f1:=x-y^2-1/4;
```

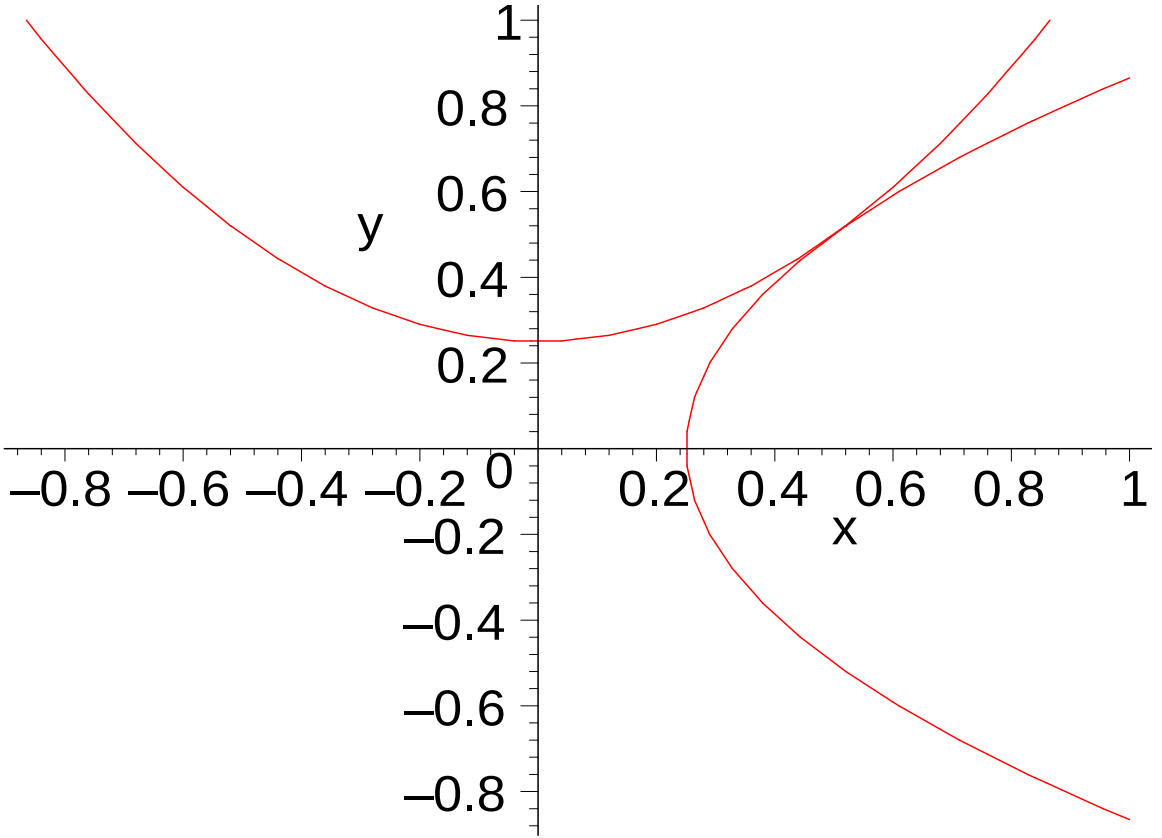
$$f1 := x - y^2 - \frac{1}{4}$$

```
> f2:=x^2-y+1/4;
```

$$f2 := x^2 - y + \frac{1}{4}$$

má právě jedno řešení $\mathbf{x}_1 = [0,5; 0,5]$, což lze ověřit jednoduchým dosazením nebo graficky.

```
> plots[implicitplot]({f1=0,f2=0},x=-1..1,y=-1..1);
```




```
> reseni:=fsolve({f1,f2},{x=0.3,y=0.5});
```

$$reseni := \text{fsolve}(\{x - y^2 - \frac{1}{4},\ x^2 - y + \frac{1}{4}\},\ \{y = 0.5,\ x = 0.3\})$$

Maple tedy zadanou úlohu příkazem `fsolve` nevyřešil. V příkladech pro tento text naleznete řešení tohoto příkladu Newtonovou metodou.

Newtonova metoda

Nejdříve zopakujeme odvození Newtonovy iterační metody pro soustavu dvou nelineárních rovnic (2). Dále v této části ukážeme, jak postupovat pomocí systému Maple při řešení konkrétní soustavy nelineárních rovnic.

Stejně jako jiné iterační metody vychází Newtonova metoda z vhodného bodu $\mathbf{X_0} \in \mathbb{R}^2$, tzv. počátečního přiblížení, a sestrojuje posloupnost iterací $\mathbf{X_i}$, která pro $i \to \infty$ konverguje k hledanému řešení soustavy (2).

Nechť $\mathbf{X_0} = [x_0, y_0]$ je počáteční přiblížení k řešení $\mathbf{X} = [x, y]$ soustavy (2). Položíme

$$\mathbf{X} = \mathbf{X}_0 + \Delta\mathbf{X}, \quad \text{t}j. \quad x = x_0 + \Delta_x\,,\ y = y_0 + \Delta_y\,,$$

a budeme požadovat, aby $\mathbf{f}(\mathbf{X}) = \mathbf{0}$, tedy aby

$$\begin{aligned} f_1(x_0 + \Delta_x, y_0 + \Delta_y) &= 0 \\ f_2(x_0 + \Delta_x, y_0 + \Delta_y) &= 0\,. \end{aligned} \tag{3}$$

Napíšeme Taylorův rozvoj pro vektorovou funkci $\mathbf{f} = (f_1, f_2)$ v okolí bodu $\mathbf{X_0}$, kde se omezíme pouze na členy lineární v $\Delta\mathbf{X}$. Použijeme tedy Taylorův rozvoj prvního stupně pro funkce f_1 a f_2 a dostáváme

přibližně

$$\begin{aligned} f_1(\mathbf{X_0}) + \frac{\partial f_1}{\partial x}(\mathbf{X_0})\,\Delta_x + \frac{\partial f_1}{\partial y}(\mathbf{X_0})\,\Delta_y &= 0 \\ f_2(\mathbf{X_0}) + \frac{\partial f_2}{\partial x}(\mathbf{X_0})\,\Delta_x + \frac{\partial f_2}{\partial y}(\mathbf{X_0})\,\Delta_y &= 0\,. \end{aligned} \tag{4}$$

Tuto soustavu lineárních algebraických rovnic pro neznámé Δ_x a Δ_y přepíšeme do tvaru

$$\begin{bmatrix} \frac{\partial f_1}{\partial x}(\mathbf{X_0}) & \frac{\partial f_1}{\partial y}(\mathbf{X_0}) \\ \frac{\partial f_2}{\partial x}(\mathbf{X_0}) & \frac{\partial f_2}{\partial y}(\mathbf{X_0}) \end{bmatrix} \begin{bmatrix} \Delta_x \\ \Delta_y \end{bmatrix} = - \begin{bmatrix} f_1(\mathbf{X_0}) \\ f_2(\mathbf{X_0}) \end{bmatrix},$$

tj.

$$\mathbf{J}(\mathbf{X_0}) \cdot \Delta\mathbf{X} = -\mathbf{f}(\mathbf{X_0}), \tag{5}$$

kde $\mathbf{J}(\mathbf{X_0})$ je matice parciálních derivací vektorové funkce $\mathbf{f}$ v bodě $\mathbf{X_0}$, tedy Jacobiho matice. Je-li matice $\mathbf{J}(\mathbf{X_0})$ regulární, má soustava (5) právě jedno řešení $\Delta\mathbf{X}$. Označíme-li

$$\mathbf{X_1} = \mathbf{X_0} + \Delta\mathbf{X}\,, \tag{6}$$

je $\mathbf{X_1}$ první další přiblížení soustavy nelineárních rovnic (2).

Pro výpočet další iterace, tj. další aproximace kořene soustavy (2), postupujeme stejným způsobem: považujeme $\mathbf{X_1}$ za nové počáteční přiblížení a nalezneme řešení soustavy lineárních algebraických rovnic (jako v případě rovnice (5)) pro nový vektor neznámých $\Delta\mathbf{X}$. Obdobně jako v (6) dostáváme další aproximaci řešení $\mathbf{X_2} = \mathbf{X_1} + \Delta\mathbf{X}$.

Dále ukážeme konkrétní postup při řešení soustavy nelineárních rovnic typu (2) pomocí systému Maple.

Z předchozí diskuse v tomto textu víme, že pro určení počátečního přiblížení je vhodné v systému Maple znázornit nulové vrstevnice funkcí f_1 a f_2 příkazem `implicitplot`.

Při hledání další aproximace je numericky nejobtížnější řešit soustavu lineárních algebraických rovnic (LAR). Navíc obdobnou soustavu LAR musíme řešit opakovaně pro každou další iteraci. Mohli bychom sice (tedy zejména pro $n = 2$) pro výpočet řešení soustavy LAR použít Cramerova pravidla, ale výpočet determinantů vyžaduje vždy velké množství operací. Obecně je numericky výhodné řešit soustavu lineárních algebraických rovnic pomocí již osvědčených metod. Většinou se pro soustavu lineárních algebraických rovnic používá některý ze softwarových "balíků" s chybovým hlášením, pokud je matice soustavy "blízká" matici singulární.

Jelikož máme k dispozici systém Maple, načteme hned na začátku příkazem `with(linalg)` programy umožňující řešit úlohy lineární algebry, tedy také příkazem `linsolve` řešit soustavu lineárních algebraických rovnic.

```
> with(linalg):

Warning, the protected names norm and trace have been redefined
and unprotected
```

Příklad 4. Je dána soustava nelineárních rovnic

```
> f1:=x^3+y^3-8;
```

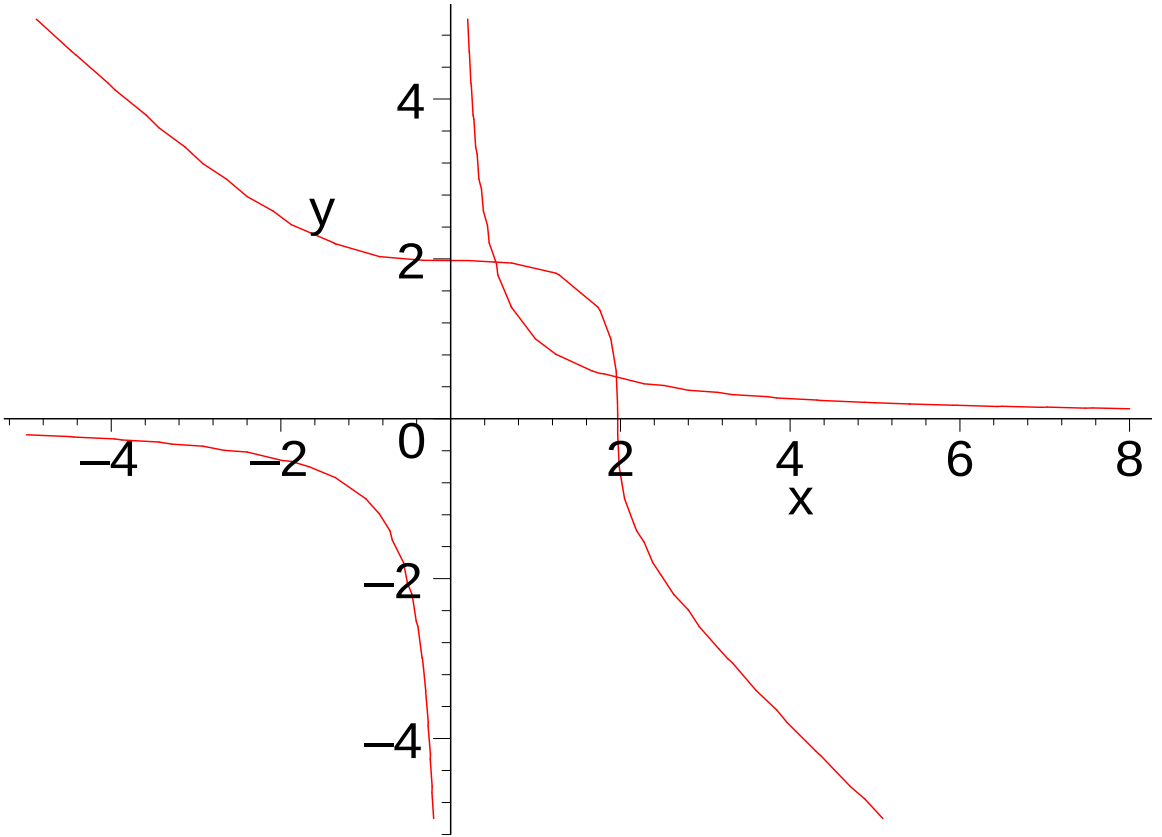
$$f1 := x^3 + y^3 - 8$$

```
> f2:=y*x-1;
```

$$f2 := y\,x - 1$$

Grafy nulových vrstevnic se protínají ve dvou bodech.

```
> plots[implicitplot]({f1=0,f2=0},x=-5..8,y=-5..5);
```



```
> f := vector([f1,f2]);
```

$$f := [x^3 + y^3 - 8,\ y\,x - 1]$$

Budeme potřebovat Jacobiho matici, tj. matici parciálních derivací vektorové funkce $\mathbf{f}$. V Maplu existuje funkce `jacobian` pro její výpočet:

```
> Df := jacobian(f,[x,y]);
```

$$Df := \begin{bmatrix} 3\,x^2 & 3\,y^2 \\ y & x \end{bmatrix}$$

Pomocí obrázku znázorňující nulové vrstevnice zvolíme počáteční aproximaci jednoho z kořenů, například $Aprox_0 = [1; 2]$. Pro dosazení jednotlivých složek bodu $Aprox_0$ do Jacobiho matice Df použijeme příkazu `subs`. Poznamenejme, že příkaz `evalm` vyhodnocuje výrazy ve vektorech nebo maticích. Následujícími příkazy získáme první aproximaci kořene soustavy $Aprox_1$.

```
> Aprox[0]:=[1.0,2.0];
```

$$Aprox_0 := [1.0,\ 2.0]$$

```
> J[0]:=subs(x=Aprox[0][1],y=Aprox[0][2],evalm(Df));
```

$$J_0 := \begin{bmatrix} 3.00 & 12.00 \\ 2.0 & 1.0 \end{bmatrix}$$

```
> F[0]:=subs(x=Aprox[0][1],y=Aprox[0][2],evalm(f));
```

$$F_0 := [1.000,\ 1.00]$$

```
> DX:=linsolve(J[0],-F[0]);
```

$$DX := [-0.5238095238,\ 0.04761904767]$$

```
> Aprox[1]:=evalm(Aprox[0]+DX);
```

$$Aprox_1 := [0.4761904762,\ 2.047619048]$$

Další aproximace kořene bychom získali opakováním stejných příkazů, jenom s jinými hodnotami. Pro takový případ se ve všech programovacích jazycích používá příkaz cyklu. V systému Maple je rovněž možno použít příkaz cyklu, který má tvar

```
for n from 1 to nmax do
```

pro opakující se skupinu příkazů za výrazem `do`. Nejdříve se dosadí `n=1` a po provedení skupiny příkazů (za příkazem `do`) se hodnota `n` zvětší o `1`. Skupina příkazů se provede znovu, a to tolikrát, až `n` dosáhne hodnotu `n=nmax`, tj. celkem nmax-krát. Pokud například chceme vypočítat další tři aproximace řešení Newtonovou metodou, zvolíme $nmax = 3$.

```
> nmax:=3;
```

$$nmax := 3$$

```
> for n from 1 to nmax do
> J[n]:=subs(x=Aprox[n][1],y=Aprox[n][2],evalm(Df));
> F[n]:=subs(x=Aprox[n][1],y=Aprox[n][2],evalm(f));
> DX:=linsolve(J[n],-F[n]);
> Aprox[n+1]:= evalm(Aprox[n]+DX);
> end do;
```

$$J_1 := \begin{bmatrix} 0.6802721088 & 12.57823130 \\ 2.047619048 & 0.4761904762 \end{bmatrix}$$

$$F_1 := [0.693121698,\ -0.0249433105]$$

$$DX := [0.02531510182,\ -0.05647398579]$$

$$Aprox_2 := [0.5015055780,\ 1.991145062]$$

$$J_2 := \begin{bmatrix} 0.7545235344 & 11.89397597 \\ 1.991145062 & 0.5015055780 \end{bmatrix}$$

$$F_2 := [0.020343096,\ -0.0014296448]$$

$$DX := [0.001167441817,\ -0.001784429226]$$

$$Aprox_3 := [0.5026730198,\ 1.989360633]$$

$$J_3 := \begin{bmatrix} 0.7580404944 & 11.87266718 \\ 1.989360633 & 0.5026730198 \end{bmatrix}$$

$$F_3 := [0.000021070,\ -0.20831\,10^{-5}]$$

$$DX := [0.1520067082\,10^{-5},\ -0.1871716950\,10^{-5}]$$

$$Aprox_4 := [0.5026745399,\ 1.989358761]$$

Vidíme, že se $Aprox_4$ liší od $Aprox_3$ až na pátém desetinném místě. Pro praktické účely můžeme tedy prohlásit kořen $Aprox_4$ za přibližný kořen soustavy. Většinou je však předem zadána přesnost ε , s jakou požadujeme kořen soustavy. Potom ukončíme proces výpočtu aproximací až tehdy, když platí

$\|\Delta\mathbf{X}\| < \varepsilon$. Poznamenejme, že je vhodné použít numericky nejjednodušší normu vektoru, tedy ověřit, zda-li již platí $|\Delta_x| + |\Delta_y| < \varepsilon$.

Závěrem nutno říci, že Newtonova metoda je účinná jen tehdy, pokud konverguje. Nalezení vhodného počátečního přiblížení je často obtížné, ale pokud metoda konverguje, konverguje ke kořeni soustavy.

Pro soustavy nelineárních rovnic nejsou obecně známy metody, které vždy konvergují. Problém řešení soustav nelineárních rovnic je jedním z nejobtížnějších problémů v numerické matematice a v současnosti nejsou známy obecné uspokojivé postupy pro jeho řešení.