Nonlinear equations

Robert Mařík

27. listopadu 2010

1 Nonlinear equations

We solve $\cos(x) = x$ or equivalently $\cos(x) - x = 0$ using several different methods.

- When solving $\cos(x) = x$ we look for fixed point of the function $f : x \mapsto \cos(x)$.
- When solving $\cos(x) - x = 0$ we look for x-intercept of the function $g : x \mapsto \cos(x) - x$.
- The value x_0 is a solution with correct 300 decimal places (from Wolfram Alpha).

Sage code

```
f(x)=cos(x)
g(x)=cos(x)-x
x0=0.7390851332151606416553120876738734040134117589007574649656806357732846548835475945993761069317
print x0
```

```
0.739085133215160641655312087673873404013411758900757464965680635773284654883547594599
93761069317665318498012466439871630277149036913084203157804405746207786885249038915392
8943884509523480133563127677223158095635377657245120437341993643351253840978003434064
67004794021434780802718018837711361382042066316335037277991 70
```

Remark: The methods presented in this article exhibit slow convergence if looking for a root x_0 of a function $g(x)$ which satisfies $g(x_0) = 0 \approx g'(x_0)$ (or equivalently for fixed point x_0 of the function $f(x)$ which satisfies $f(x_0) = x_0$ and $|f'(x)| \approx 1$) and thus the methods are not suitable for looking for repeated roots. Fortunately, this is not the case for equation $\cos(x) - x = 0$.

2 Bisection

This is the simplest method to find approximate solution of $g(x) = 0$ if the function $g(x)$ changes its sign. It is based on Bolzano's intermediate value theorem and explained at Wikipedia.

Sage code

```
a,b=0,1
for i in range(10):
    x=(a+b)/2
    val=n(g(x))
    left_point=n(g(a))
    if left_point*val<0:
        b=x
    else:
        a=x
x=n((a+b)/2)
print "Value:",x,"\nError:",(x-x0)
```

```
Value: 0.738769531250000
Error: -0.000315601965160672
```

[0]Podporováno grantem FRVŠ 131/2010.

3 Banach theorem

Based on fixed point theorem we solve $f(x) = x$. The iteration scheme is $x_{n+1} = f(x_n)$ and under some assumptions, such as $|f'(x)| \leq K < 1$, it converges to the solution of $f(x) = x$.

Sage code

```
x=1
for i in range(10):
    x=n(f(x),100)
print "Value:",x,"\nError:",(x-x0)
```

```
Value: 0.74423735490055686343348005215
Error: 0.0051522216853962217781679644769
```

4 Newton's method

Iteration scheme $x_{n+1} = x_n - \dfrac{g(x_n)}{g'(x_n)}$ converges to the solution of $g(x) = 0$. The method has simple geometric interpretation explained at Wikipedia.

Sage code

```
gder(x)=diff(g(x),x)
x=1
for i in range(3):
    x=n(x-g(x)/gder(x),digits=50)
print "Value:",x,"\nError:",(x-x0).n()
```

```
Value: 0.73908513338528396976012512085680433288953312317019
Error: 1.70123328104813e-10
```

5 Halley's method

Extension of Newton's mehotd. The iteration scheme is $x_{n+1} = x_n - \dfrac{2g(x_n)g'(x_n)}{2\big(g'(x_n)\big)^2 - g(x_n)g''(x_n)}$. Explanation for this formula is available at Mathwolrd.

Sage code

```
gder(x)=diff(g(x),x)
gder2(x)=diff(g(x),x,2)
x=1
for i in range(3):
    x=n(x-2*g(x)*gder(x)/(2*gder(x)^2-g(x)*gder2(x)),digits=50)
print "Value:",x,"\nError:",(x-x0).n()
```

```
Value: 0.73908513321516064165531208770754384945010360794224
Error: 3.36704454366919e-29
```

6 Householder's Method

Extension of previous mehotds. The iteration scheme is $x_{n+1} = x_n - \dfrac{g(x_n)}{g'(x_n)}\left[1 + \dfrac{g(x_n)g''(x_n)}{2\big(g'(x)\big)^2}\right]$ (from Mathwolrd).

Sage code

```
gder(x)=diff(g(x),x,1)
gder2(x)=diff(g(x),x,2)
x=1
for i in range(3):
    x=n(x-g(x)/gder(x)*(1+g(x)*gder2(x)/(2*gder(x)^2)),digits=50)
print "Value:",x,"\nError:",(x-x0).n()
```

```
Value: 0.739085133215160641655312088348522013321394386319971
Error: 6.74648609307983e-28
```

Another general formula known as Householder's formula is $x_{n+1} = x_n + d\dfrac{\left(\frac{1}{f(x_n)}\right)^{(d-1)}}{\left(\frac{1}{f(x_n)}\right)^{(d)}}$, where $d \geq 1$ is an integer and $(\cdot)^{(d)}$ denotes derivative of order d. For more informations and detailed explanation see Wikipedia. Note that using $d = 5$ we have more than 250 digits correct after only 3 iterations!

Sage code

```
d=5
gder(x)=diff(1/g(x),x,d-1)
gder2(x)=diff(1/g(x),x,d)
x=1
for i in range(3):
    x=n(x+d*gder(x)/gder2(x),digits=500)
print "Value:",x,"\nError:",(x-x0).n()
```

```
Value: 0.73908513321516064165531208767387340401341175890075746496568063577328465488835
47594599376106931766531849801246643987163027714903691308420315780440574620778688524490
38915392894388450952348013356312767722315809563537765724512043734199364335125384097780
03434064670047940214348243571441861949998952632757515752444344523473987999132031534401
92729713635771549418816012633252776299707992148657698692299044341587403166317574822203
4131979812539092734705752611693411169574475037195206662195777546010604655248818562290
Error: 4.35544261673573e-270
```