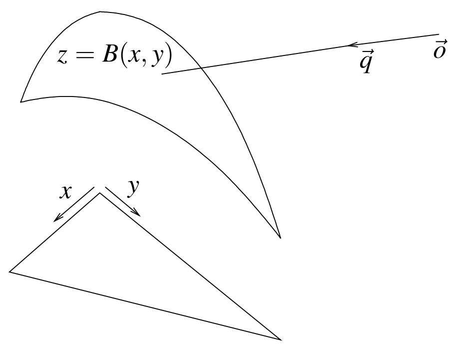
TANA09 Lecture 3

- Application - Ray tracing and Beziér surfaces.
- Error estimate. Cancellation.
- The Newton-Raphson method. Analysis.
- Order of convergence. The Secant method.
- Single and double roots.
- Application - Implementation of the square root.

Application - Ray Tracing and Bézier surfaces



A ray of light originates from $\vec{o}$ and points in the direction $\vec{q}$.

Where does the line $p(t) = \vec{o} + t\vec{q}$ intersect the surface $z = B(x, y)$?

Algorithm We solve the problem by the steps

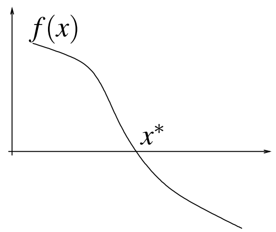
- The surface patch associated with a certain polygon is given by a cubic polynomial.
$$z = B(x, y),$$
where x, y, and z are the *local coordinates*.
- Write the vectors $\vec{o}$ och $\vec{p}$ in the *local coordinates*.
- Find the root of the polynomaial

$$f(t) = o_z + tq_z - B(o_x + tq_x, o_y + tq_y) = 0.$$

Spline surfaces and curves is the last exercise of the course.

Masters thesis Joakim Löv, 2006. Implementation on a GPU.

Error Estimate



Question A numerical method generates a sequence $x_k \to x^*$, where x^* is a root of the equation $f(x) = 0$. What is a suitable stopping criteria?

Let $\bar{x}$ be an approximation of the root x^*. Can we estimate the error $|\bar{x} - x^*|$?

Lemma Let $\bar{x}$ be an approximation of the root x^*. Then

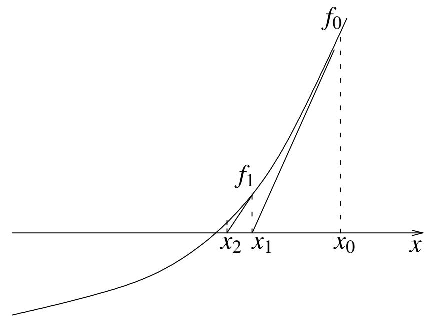
$$|\bar{x} - x^*| \leq \frac{|f(\bar{x})|}{M},$$

where $|f'(x)| \geq M$ near the root x^*.

Exemple We have found an approximate root $\bar{x} = 0.56714335$ to the equation $f(x) = x - \mathrm{e}^{-x}$. Estimate the error.

Remark Usually we have *cancellation* in the computation of the function value $f(\bar{x})$. If $|f(\bar{x})| \approx \mu$ we cannot neglect the computational errors.

The Newton-Raphson method



Newton-Raphson Given x_0 we compute a sequence

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \qquad k = 0, 1, 2 \ldots$$

Exemple Solve the equation $f(x) = x - \mathrm{e}^{-x} = 0$. Here we have $f'(x) = 1 + \mathrm{e}^{-x}$.

k	x_k	$f(x_k)$
0	0.550000000	$-2.69 \cdot 10^{-2}$
1	0.567089834	$-8.38 \cdot 10^{-5}$
2	0.567143290	$-8.10 \cdot 10^{-10}$

Very fast convergence. Always convergence to a single root if the starting approximation is good enough.

Two function calls in each step (both $f(x)$ and $f'(x)$).

Requires that $f'(x)$ is available.

Order of convergence

Definition Let $\{x_k\}$ be a sequence that converges to x^*. The *order of convergence* is the largest integer p such that

$$\lim_{k \to \infty} \frac{|x_k - x^*|}{|x_{k-1} - x^*|^p} = C < \infty.$$

Remark If $p = 1$ the convergence is *linear* and if $p = 2$ we have *quadratic* convergence.

Example Consider the fixed point iteration $x_{k+1} = \phi(x_k) = \mathrm{e}^{-x_k}$. Show that the convergence is *linear*.

Theorem Let $\{x_k\}$ be the sequence computed by Newton-Raphsons method. If the method converges to a single root x^* then

$$|x_{k+1} - x^*| \approx C|x_k - x^*|^2,$$

Thus the order of convergence is $p = 2$.

Exemple Suppose $|x_{k+1} - x^*| \approx 3.5|x_k - x^*|^2$ och $|x_0 - x^*| < 0.5 \cdot 10^{-2}$. What does this mean in practice?

We find that $|x_1 - x^*| < 9.0 \cdot 10^{-5}, \quad |x_2 - x^*| < 2.8 \cdot 10^{-8}, \ldots$

The Secant method

If the derivative $f'(x)$ is unavailable we can approximate

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

Secant method Given x_0 and x_1 we computes a sequence

$$x_{k+1} = x_k - f(x_k)\left(\frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}\right)^{-1}.$$

Questions What is the order of comvergence? Stable?

Exemple Solve the equation $f(x) = x - \mathrm{e}^{-x} = 0$ using the Secant method.

k	x_k	$f(x_k)$
0	0.700000000000000	$2.03 \cdot 10^{-1}$
1	0.600000000000000	$5.12 \cdot 10^{-2}$
2	0.566373515585849	$-1.21 \cdot 10^{-3}$
3	0.567147844602993	$7.14 \cdot 10^{-6}$
4	0.567143291044208	$9.94 \cdot 10^{-10}$
5	0.567143290409783	$-8.9 \cdot 10^{-16}$
6	0.567143290409784	$-1.11 \cdot 10^{-16}$

Very fast convergence. How to estimate the rate of convergence p?

Lemma The rate of convergence is given by

$$p = \lim_{k\to\infty} \frac{\log(\varepsilon_k/\varepsilon_{k-1})}{\log(\varepsilon_{k-1}/\varepsilon_{k-2})}, \quad \text{where } \varepsilon_k = |x_k - x^*|.$$

Example The iterations x_2, x_3 and x_4, with $\varepsilon_k = |x_k - x_6|$, gives

$$p \approx \frac{\log(\varepsilon_4/\varepsilon_3)}{\log(\varepsilon_4/\varepsilon_3)} = 1.7308\ldots$$

Instead using x_3, x_4 and x_5 we get $p \approx 1.57$.

Theorem The order of convergence for the Secant method is $p = (1 + \sqrt{5})/2 \approx 1.618$.

Double roots

Definition A function $f(x)$ has a root x^* of *multiplicity* k if we can write

$$f(x) = (x - x^*)^k g(x), \text{ where } g(x^*) \neq 0.$$

Example For a *single root* as have $f(x^*) = 0$ and $f'(x^*) \neq 0$.

For a *double root* both $f(x^*) = 0$ and $f'(x^*) = 0$.

Question How to modify the error estimate to cover the case of double roots?

Example Solve $f(x) = (x - e^{-x})^2$ using Newtons method.

k	x_k	$f(x_k)$
0	0.550000000000000	$7.3 \cdot 10^{-4}$
1	0.581851788560507	$5.3 \cdot 10^{-4}$
2	0.555794973335401	$3.2 \cdot 10^{-4}$
3	0.576692919807852	$2.2 \cdot 10^{-4}$
4	0.559647895670487	$1.4 \cdot 10^{-4}$
5	0.57337002513594	$9.5 \cdot 10^{-5}$
10	0.565003205175893	$1.1 \cdot 10^{-5}$
20	0.566881467726078	$1.7 \cdot 10^{-7}$

This is clearly linear convergence!

Lemma If x^* is a double root then Newtons method has the order of convergence $p = 1$.

Lemma Let $\bar{x}$ be an approximation of a double root x^*. Then

$$|\bar{x} - x^*| \leq \left(\frac{2|f(\bar{x})|}{M}\right)^{1/2},$$

where $M \geq |f''(\xi)|$, $\xi \in (\bar{x}, x^*)$.

Example Let $\bar{x} = 0.56689$ be an approximate root to $f(x) = (x - e^{-x})^2$. Estimate the error $|\bar{x} - x^*|$.

Remark For double roots we get drastically slower convergence and worse error estimates. Tripple roots?

Application - The square root

Problem Implement $\sqrt{a}$ as efficiently as possible on a computer with IEEE double precision arithmetic.

- Solve the equation $f(x) = x^2 - a$ using the Newton-Raphson method.
- Pick the starting approximation x_0 and use that a is a *normalized floating point number* so $1 \leq a < 4$ and $1 \leq x < 2$.
- Determine the number of iteratioins. We want a `for`-loop and not a `while`-loop.
- Confirm that the accuracy is accaptable. We prefer a relative error $\leq \mu$.

Theorem For every $0 < x_0 < \infty$ the iteration sequence

$$x_{k+1} = \frac{1}{2}(x_k + \frac{a}{x_k})$$

is convergent and

$$\lim_{k \to \infty} x_k = \sqrt{a}.$$

Lemma The convergence is *quadratic* and

$$|x_{k+1} - \sqrt{a}| \leq \frac{1}{2}(x_k - \sqrt{a})^2.$$

The starting approximation $x_0 = 1.5$ gives an initial error $|x_0 - \sqrt{a}| \leq \frac{1}{2}$.

Alternatively use a *table*

a	$\sqrt{a}$
1.0	1.000000000000000
1.5	1.224744871391589
2.0	1.414213562373095
2.5	1.581138830084190
3.0	1.732050807568877
3.5	1.870828693386971

Now the initial error is at most $|x_0 - \sqrt{a}| < 0.12$. Larger table means fewer iterations.

simpler to write in $C++$ due to bitoperations.

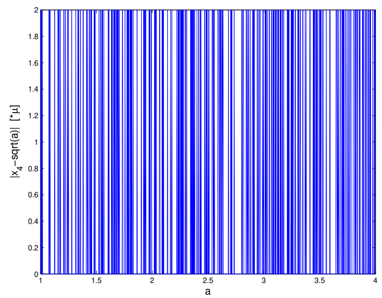
Matlab Implementation

```
function [x]=squareroot(a)
    table=[1.000000000000000
           1.224744871391589
           1.414213562373095
           1.581138830084190
           1.732050807568877
           1.870828693386971
           2.000000000000000];
    x=table(round(2*(a-1)+1));
    for k=1:4,x=(x+a/x)/2;,end;
```

A total of 4 divisions and 4 additions to compute $\sqrt{a}$.

How large is the error? There is only a finite number of floating point numbers $1 \leq a < 4$. Can test all cases!

Experimental error analysis



The difference`abs(squareroot(a)-sqrt(a))` for 1000 evenly distributed values a between 1 and 4. Equality in 751 cases and a 2μ difference in 249 cases.

In order to make a better test we need to compute $\sqrt{a}$ with extended precision or trust the mathematical analysis.

Equation solving in Matlab

The function `fzero` solves non-linear equations. It is used by typing

```
>> I = fzero( fun , x0 );
```

Exemple Solve the equation $f(x) = \mathrm{e}^{-2x} - x = 0$. In Matlab we write

```
>> f = @(x) exp(-2*x)-x;
>> x = fzero( f , 1 )
 x =
  0.4263
```

For equation solving in multiple variables there is a function `fsolve`. Considerably more difficult to prove existance and also harder to obtain rapid convergence.

The funktion `integral` computes integrals. It is used by typing

```
>> I = integral( fun , a , b );
```

Exemple Compute the integral

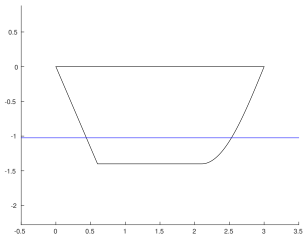
$$I = \int_0^1 (\mathrm{e}^{2x}\cos(x^2) + 4x)dx.$$

In Matlab we type

```
>> f = @(x) exp(2*x).*cos(x.^2)+4*x;
>> I = integral( f , 0 , 1)
  I =
   4.6736
```

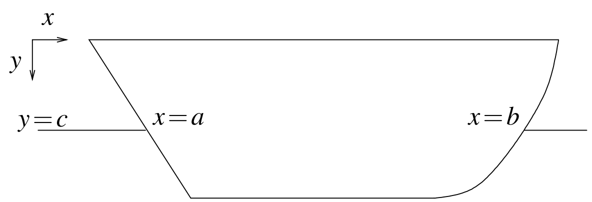

Application - How much of a boat is under water?

Problem The weight of a boat exactly corresponds to the weight of the displaced water.



Given a certain weight how can we calculate how much of the boat that is under water?

Solution Given the water line $y = c$ and the shape of the hull $y = f(x)$ we can calculate how much water is beeing displaced.



Let $g(c)$ be the displaced volume given the water level c. We have that

$$g(c) = \int_a^b (f(x) - c)dx.$$

We want to solve $g(c)\rho - L = 0$, where ρ is the densityt and L is the weight of the boat.

Matlab We have a function `ShipShape(x)` and write a function

```
function V=DisplacedVolume( c )
  a=fzero( @(x)ShipShape(x)-c ,0.4); % Root near x=0.4

  b=fzero(@(x)ShipShape(x)-c ,2.5);  % Root near 2.5

  V=integral( @(x)ShipShape(x)-c ,  a , b );
end
```

Use the function to solve the problem in Matlab

```
L=697;    % Weight in kg.
rho=998; % Density in kg/m3
c = fzero( @(c)DisplacedVolume( c )*rho-L , 0.9 )
 c=1.0253
```


Summary

- Equation solving is an important part of many applications.
- If both $f(x)$ and $f'(x)$ are known the optimal choice is almost always Newton-Raphsons method.
- The secant method is almost as fast and does not require the derivative $f'(x)$.
- Fixed point iteration is both simple and easy to use but often slow. Used when other methods are difficult to apply.