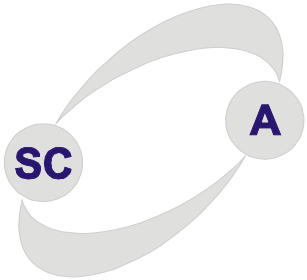
Predavanje

Predmet

Metodi optimizacije

Tema

Statička optimizacija, numeričke metode jednodimenzione optimizacije



2017 godina

Statička optimizacija, numeričke metode jednodimenzione optimizacije

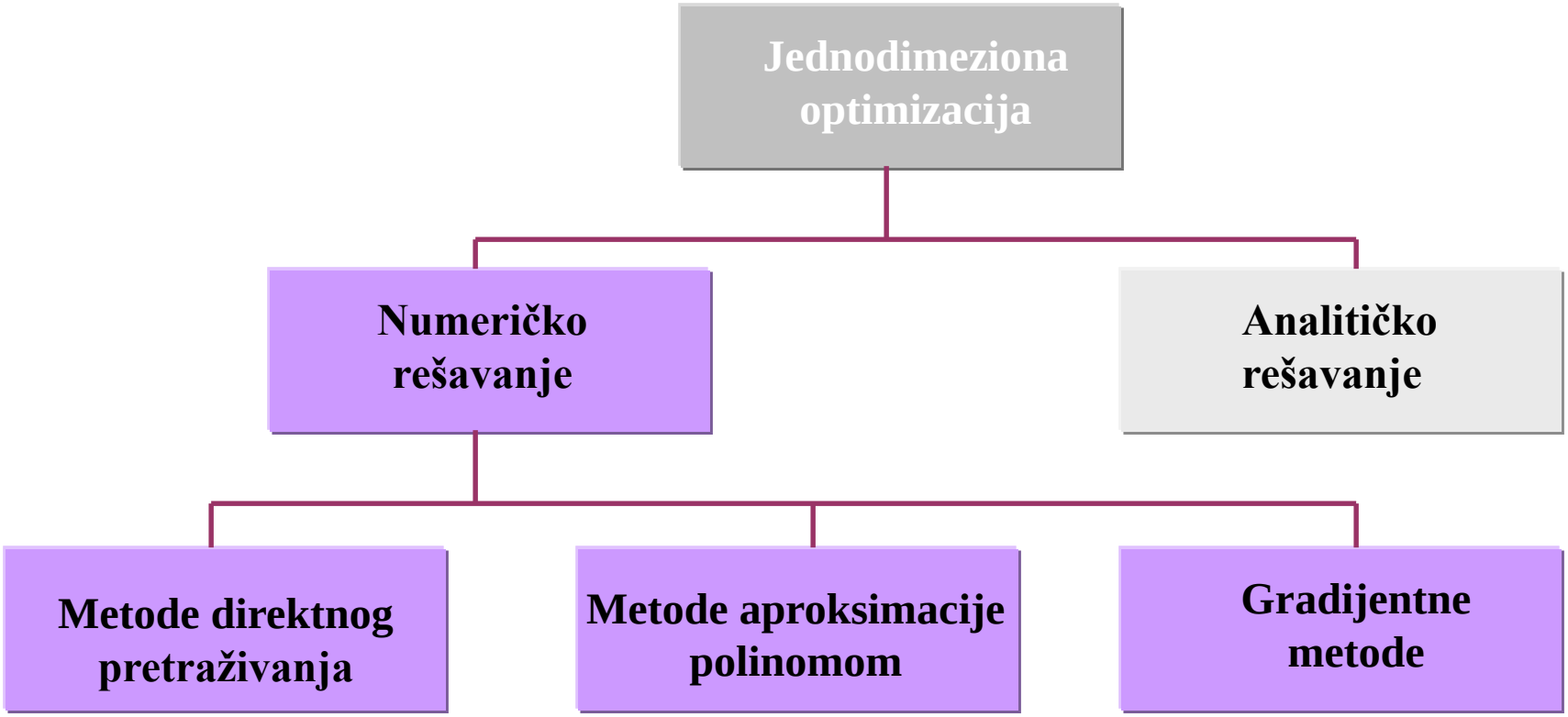
Da li su jednodimenzioni problemi laki? (Ponekad)

Da li se metodi višedimenzione optimizacije, mogu koristiti kod sistema sa jednom promenljivom? (Da)

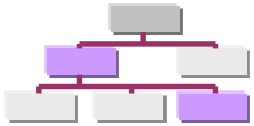
ali,

1. **Postoji određen broj važnih jednodimenzionih problema.**
2. **Predstavljaju osnovu višediemnzione numeričke optimizacije.**
3. **Jednodimenziona optimizacija je često sastavni deo složenih optimizacionih problema i softvera!**

Statička optimizacija, jednodimenziona optimizacija



Gradijentne metode



Osnovna ideja ovih metoda

Naći stacionarne tačke funkcije (f'(x)=0)

(ako je funkcija diferencijabiln do reda koji nam je potreban).

Newton-Raphson Metod

$f(x)$ *se razvija u Taylor-ov red oko* x_0

$$f(x) \approx g(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2} f''(x_0)(x - x_0)^2$$

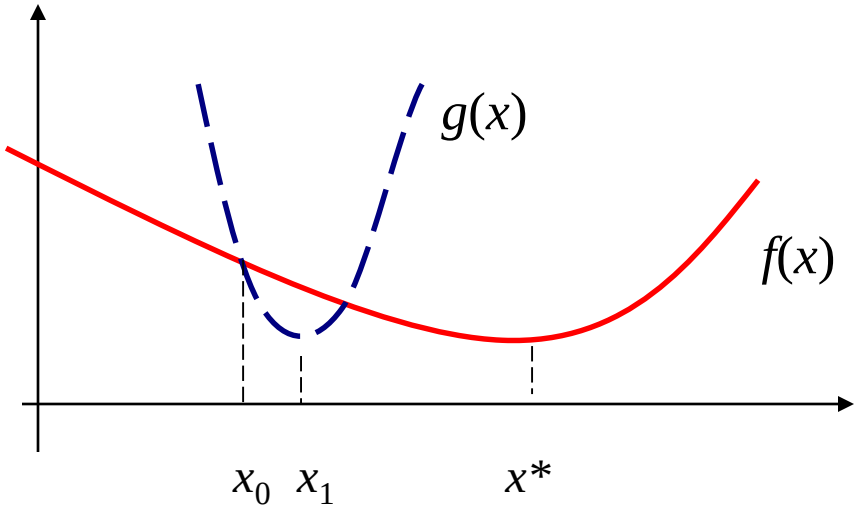
$g(x)$ *je parabola čiji je ekstrem u* x_1,

a uzima se kao polazna tačka za sledeću iteraciju

$f(x)$ *se razvija u Taylor-ov red oko* x_0

$$f(x) \approx g(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2} f''(x_0)(x - x_0)^2$$

$g(x)$ *je parabola čiji je ekstrem u* x_1,
a uzima se kao polazna tačka za sledeću iteraciju



$$g'(x_1) = 0$$

$$f'(x_0) + f''(x_0)(x_1 - x_0) = 0$$

$$x_1 = x_0 - \frac{f'(x_0)}{f''(x_0)}$$

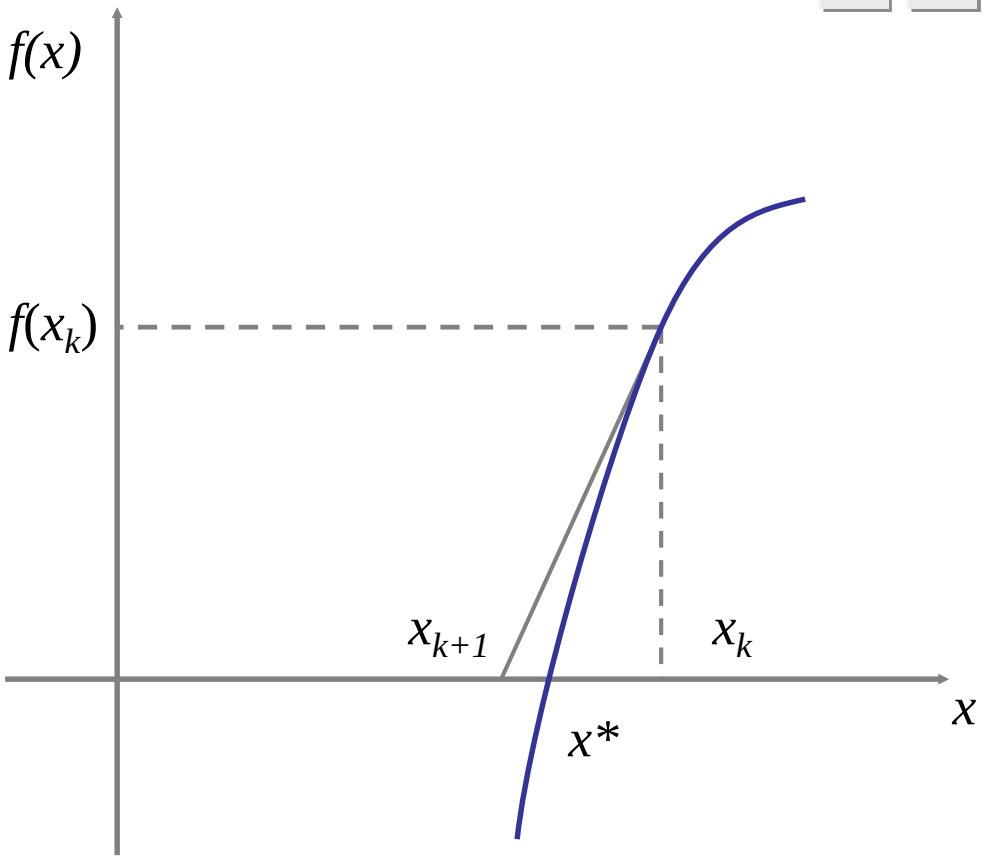
$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$$

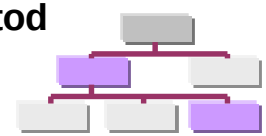
Newton-Raphson Metod

$$f'(x_k) = \frac{f(x_k)}{x_k - x_{k+1}}$$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

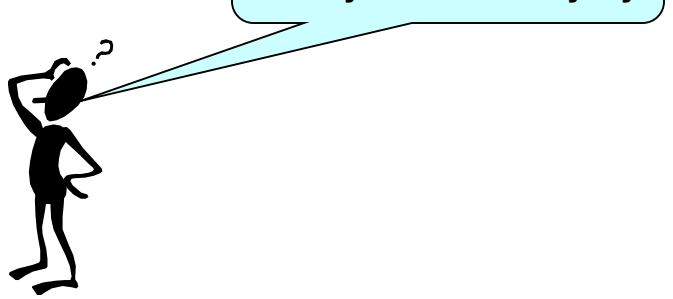
$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$$





```
function [x,fx,n]=newton(fun,dfun,d2fun,a,tol)
x = Inf;
n = 0;
while abs(x - a) > tol
   x = a;
   a = x - feval(dfun,x)/feval(d2fun,x);
   n = n + 1;
end
x = a;
fx = feval(fun,x);
```

- Početno pogađanje $$x_1 = x_0 - \frac{f'(x_0)}{f''(x_0)}$$
- Iterativni postupak $$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$
- Kriterijum zaustavljanja $$\varepsilon_{n+1} = \left| \frac{x_{n+1} - x_n}{x_{n+1}} \right|$$



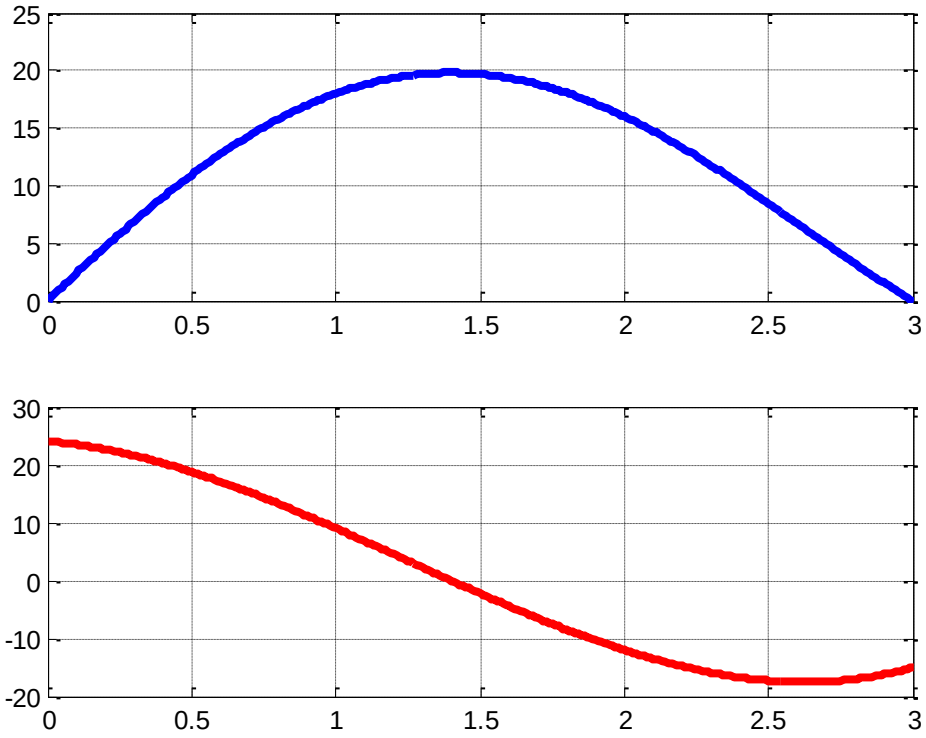
Primer

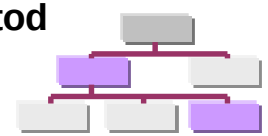
$$f(x) = x^4 - 5x^3 - 2x^2 + 24x$$

$$f'(x) = 4x^3 - 15x^2 - 4x + 24$$

$$f''(x) = 12x^2 - 30x - 4$$

$$x = [0,3]$$

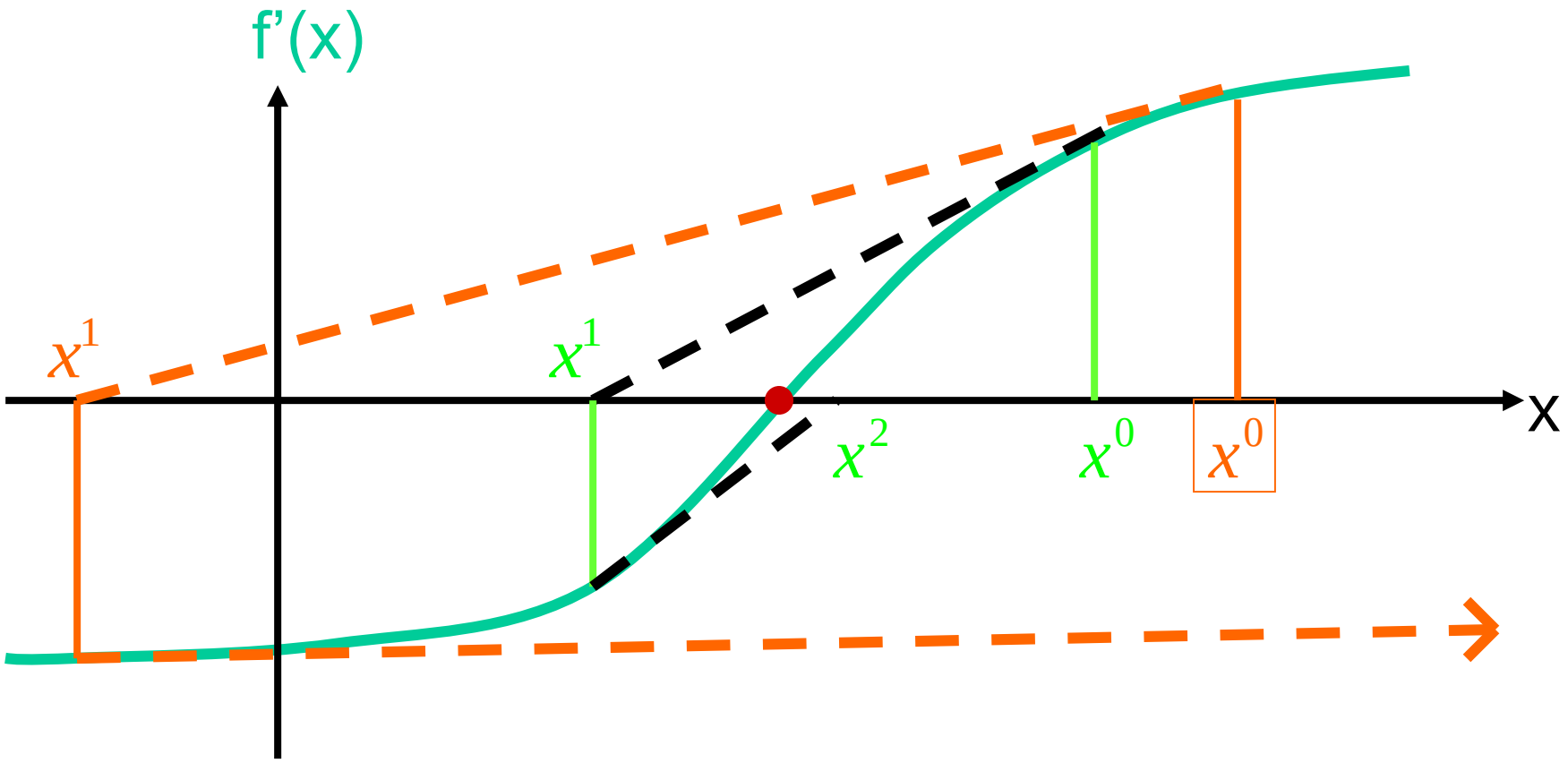




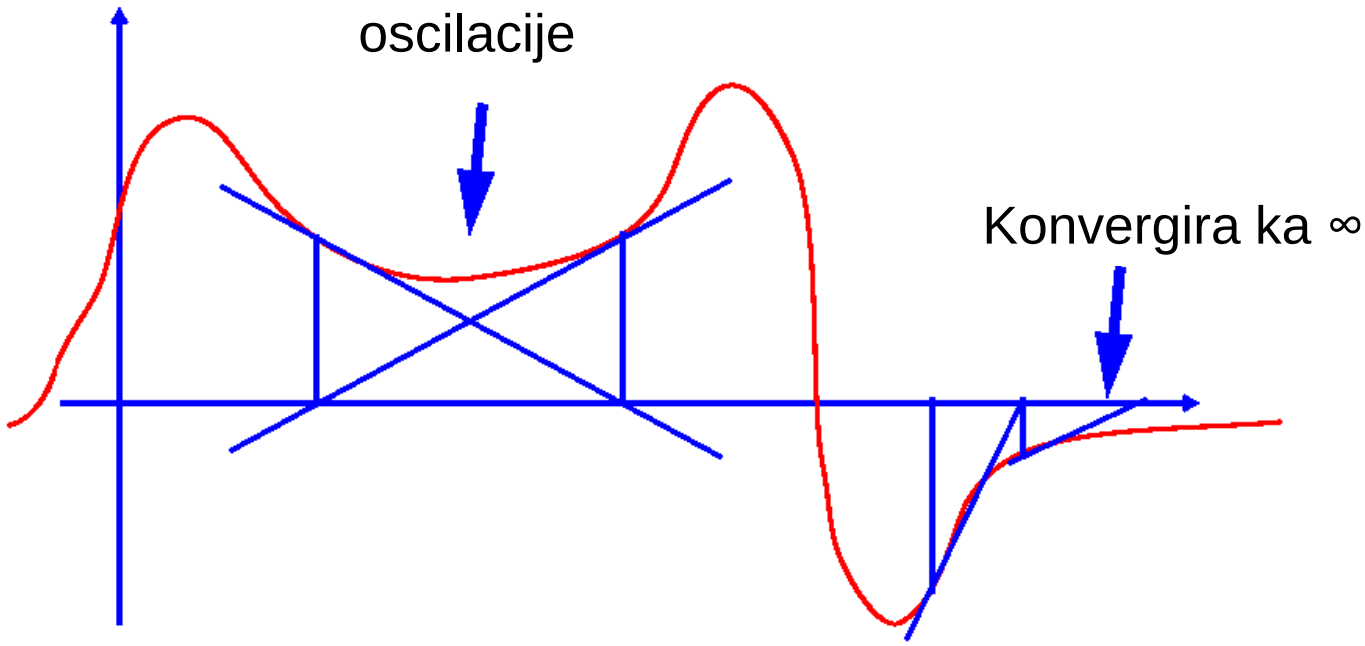
Primer

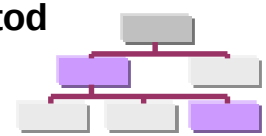
```
[x,fx,n,rez] = newton( 'fun', 'dfun1', 'd2fun1', 1, 0.0001 )
x =

    1.3989

fx =

   19.8016

n =

     3

rez =

    1.0000    1.0000    9.0000  -22.0000    0.4091    1.4091
    2.0000    1.4091   -0.2282  -22.4463   -0.0102    1.3989
    3.0000    1.3989    0.0002  -22.4839    0.0000    1.3989
```

Konvergencija zavisi od dobrog početnog pogađanja



Konvergencija zavisi od dobrog početnog pogađanja





domaći

$$f(x)' = (x-1)^3 = 0$$

$$f(x)' = x^3 - 0.03x^2 + 2.4\text{x}10^{-6} = 0$$

$$f(x)' = Sin\ x = 0$$

$$f(x)' = x^2 + 2 = 0$$

Gradijentne metode

Metod Sečice

- Ako se drugi izvod zameni konačnom razlikom,

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

Newton-va metoda postaje

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

- tačke x_0, x_1 su proizvoljne

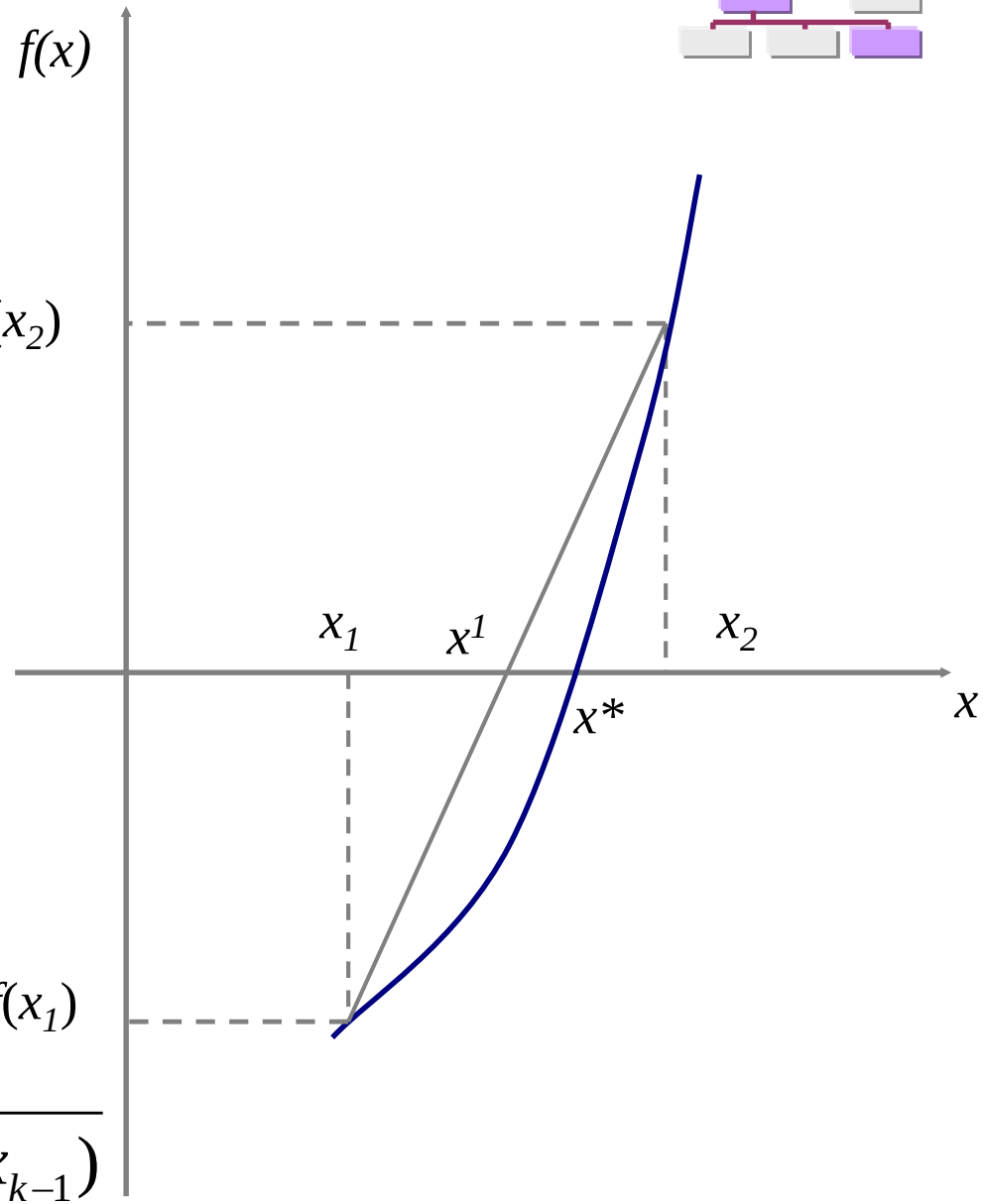
Metod Sečice

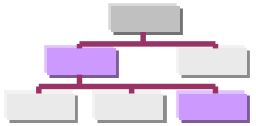
$$\frac{x^1 - x_1}{-f(x_1)} = \frac{x_2 - x_1}{f(x_2) - f(x_1)}$$

$$x^1 = x_1 - \frac{x_2 - x_1}{f(x_2) - f(x_1)} f(x_1)$$

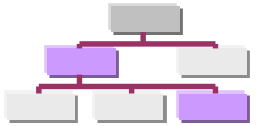
$$x^1 = x_1 - \frac{x_2 - x_1}{f'(x_2) - f'(x_1)} f'(x_1)$$

$$x_{k+1} = x_k - f'(x_k) \frac{x_k - x_{k-1}}{f'(x_k) - f'(x_{k-1})}$$





```
function [x,fx,n]=secica(fun,dfun,a,b,tol)
x = a;
a = Inf;
dfb = feval( dfun, b );
n = 0;
while abs(x - a) > tol
  a = x;
  dfa = feval( dfun, a );
  x = a - dfa * (a - b) / (dfa - dfb);
  dfb = dfa;
  b = a;
  n = n + 1;
end
fx = feval( fun, x );
```



- Početno pogađanje $x_2 = x_1 - f'(x_1)\dfrac{x_1 - x_0}{f'(x_1) - f'(x_0)}$
- Iterativni postupak $x_{k+1} = x_k - f'(x_k)\dfrac{x_k - x_{k-1}}{f'(x_k) - f'(x_{k-1})}$
- Kriterijum zaustavljanja $\varepsilon_{n+1} = \left|\dfrac{x_{n+1} - x_n}{x_{n+1}}\right|$

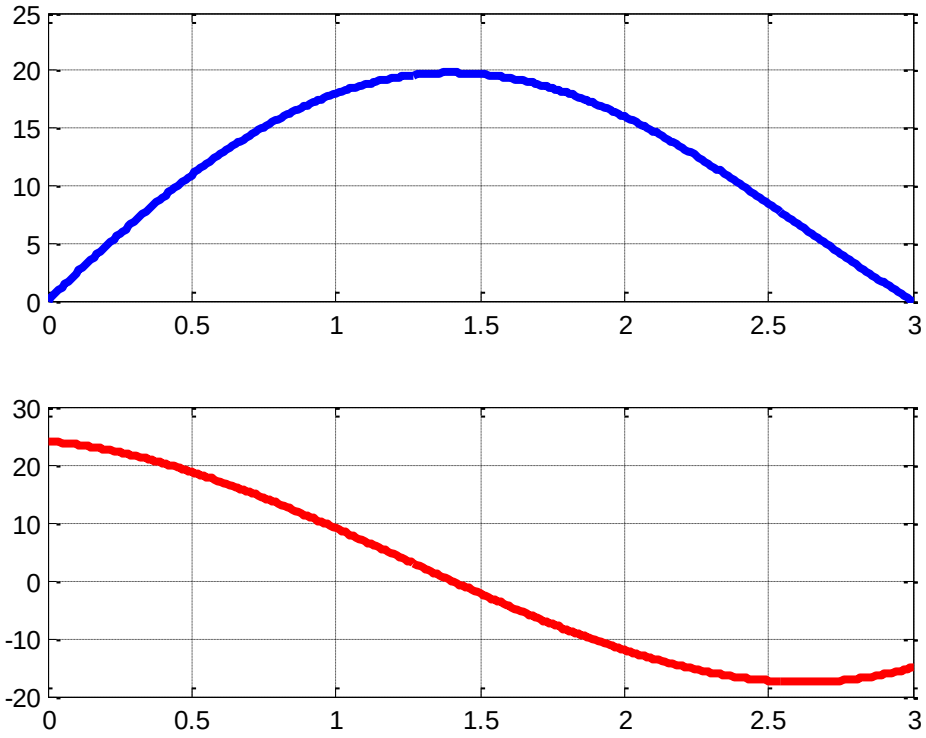
Primer

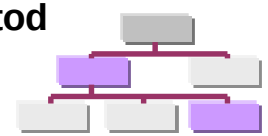
$$f(x) = x^4 - 5x^3 - 2x^2 + 24x$$

$$f'(x) = 4x^3 - 15x^2 - 4x + 24$$

$$f''(x) = 12x^2 - 30x - 4$$

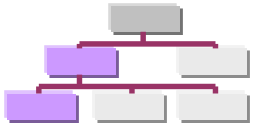
$$x = [0,3]$$





```
[x,fx,n,rez] = secica( 'fun', 'dfun1', 0,3, 0.0001 )

x =

    1.3989

fx =

   19.8016

n =

     5

rez =

    1.0000         0    3.0000   24.0000  -15.0000    1.8462    1.8462
    2.0000    1.8462         0   -9.3400   24.0000   -0.5172    1.3290
    3.0000    1.3290    1.8462    1.5805   -9.3400    0.0749    1.4038
    4.0000    1.4038    1.3290   -0.1098    1.5805   -0.0049    1.3990
    5.0000    1.3990    1.4038   -0.0005   -0.1098   -0.0000    1.3989
```

Metode direktnog pretraživanja

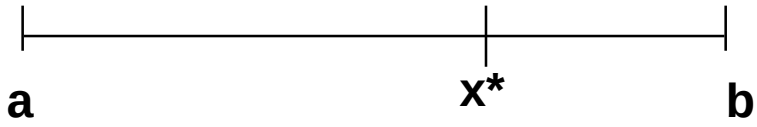


Metod direktnog pretraživanja su u osnovi metode jednodimenzione optimizacije

Smatraju ih "kičmom" nelinearnih optimizacionih algoritama

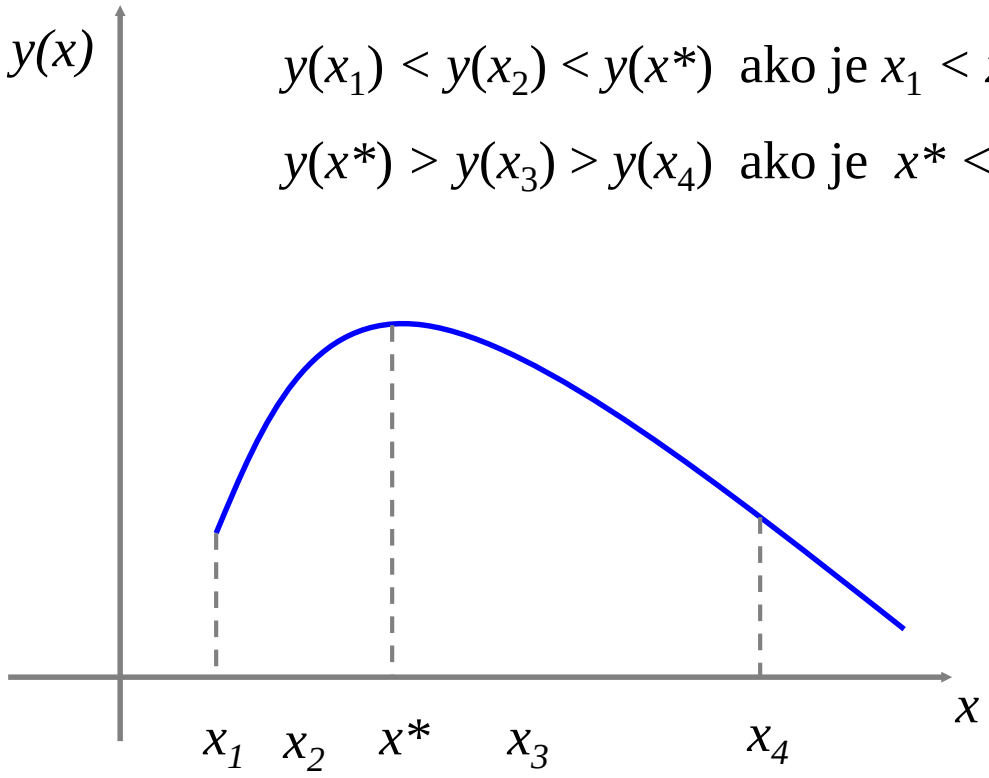
Svode se na pretraživanje zatvorenog intervala

Često pretpostavljamo da je funkcija unimodalna



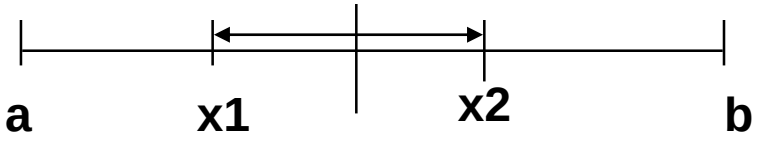
Detaljno pretraživanje zahteva $N = (b-a)/\varepsilon + 1$ proračuna u interavalu sa slike, pri čemu je ε rezolucija.

unimodalnost



$y(x_1) < y(x_2) < y(x^*)$ ako je $x_1 < x_2 < x^*$

$y(x^*) > y(x_3) > y(x_4)$ ako je $x^* < x_3 < x_4$



Pretraga da bi se našao **min** $f(x)$:

0) pretpostaviti interval [a,b]

1) naći po nekoj formuli (npr x1 = a + (b-a)/2 - ε/2 i x2 = a+(b-a)/2 + ε/2 gde je ε rezolucija).

2) porediti f(x1) i f(x2)

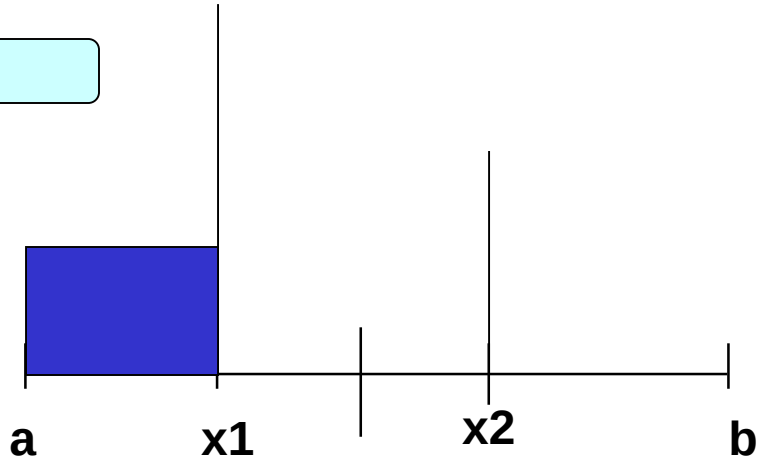
3) Ako je f(x1) < f(x2) tada eliminišemo x > x2 i b = x2

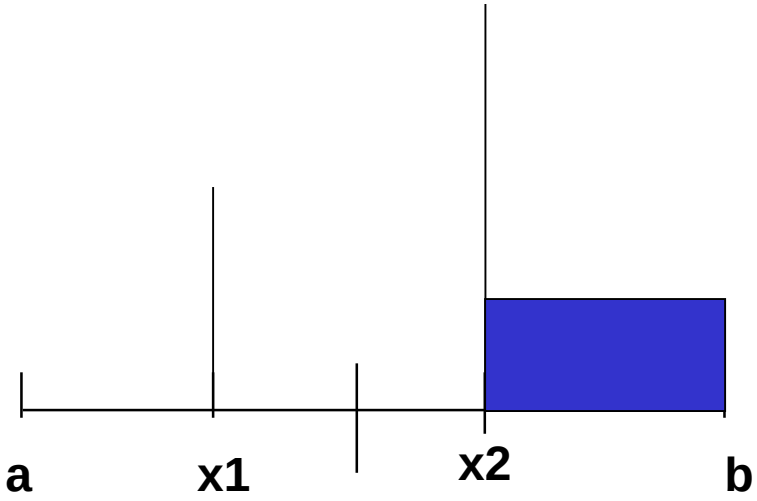
Ako f(x1) > f(x2) tada eliminišemo x < x1 i a = x1

Ako f(x1) = f(x2) tada biramo novi par tačaka

4) Nastaviti dok interval ne bude < 2 ε

Koju oblast eliminišemo





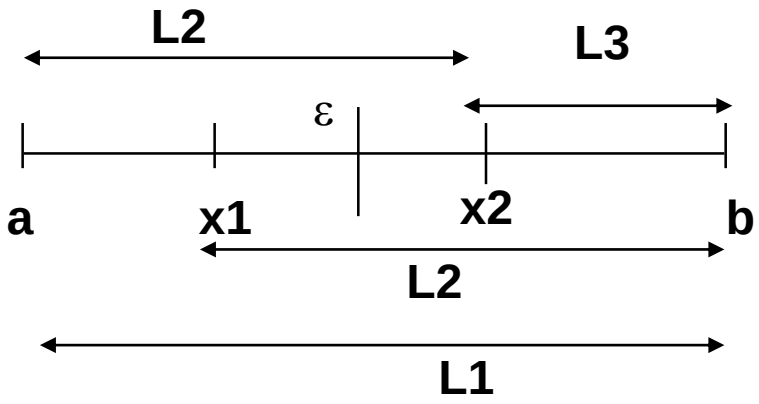
Metod direktnog pretraživanja

Fibonačijev Metod

Fibonačijevi brojevi:

1,1,2,3,5,8,13,21,34,.. odnosno , suma dva prethodna

$F_n = F_{n-1} + F_{n-2}$



L1 = L2 + L3

Može se pokazati

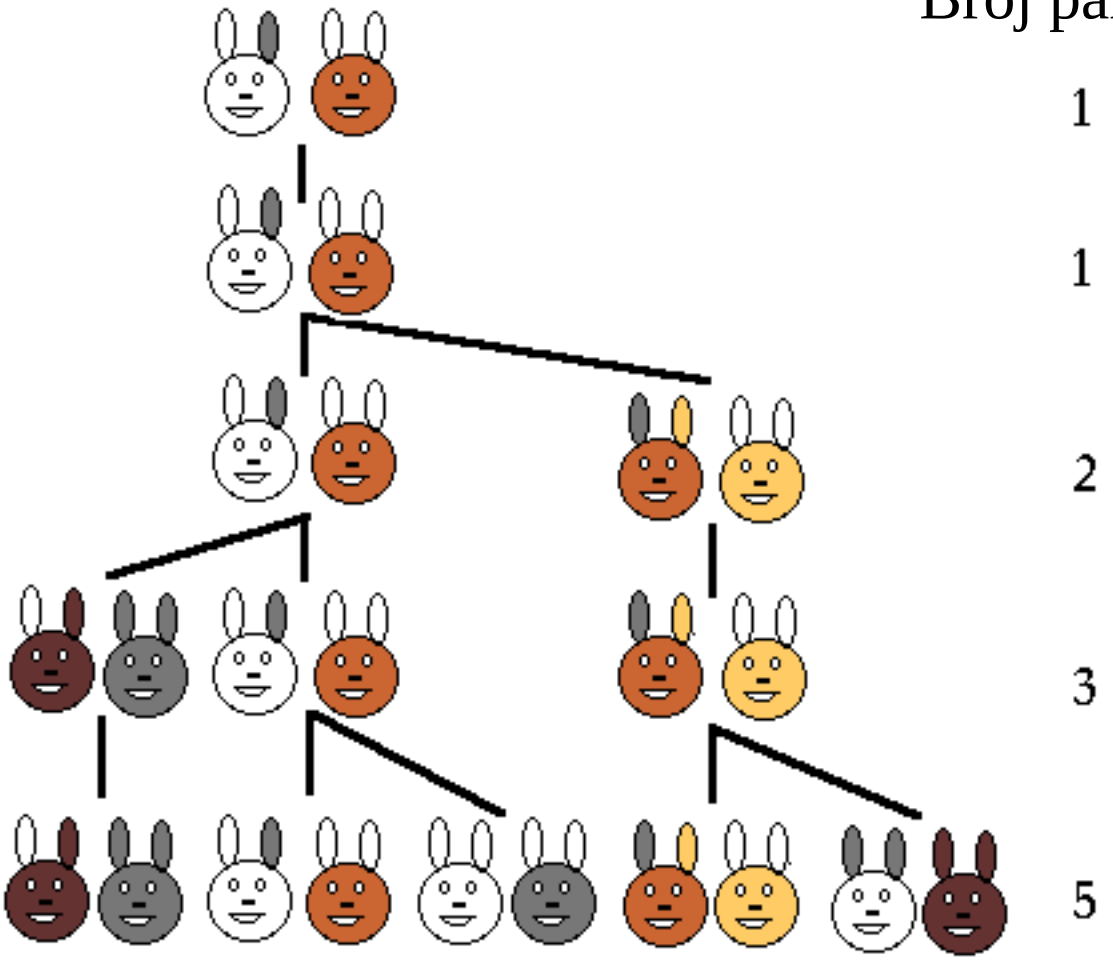
$L_n = (L1 + F_{n-2}\,\varepsilon) / F_n$

1200 te...

Fibonači je postavio slično pitanje…

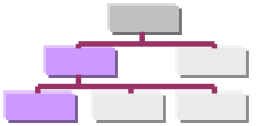
Leonardo Pisano (1170-1250)

Pretpostavimo da se tek rođeni par, mužjak i ženka, zečeva stavi u polje. Zečevi su u stanju da se razmnožavaju posle mesec dana i na kraju drugog meseca ženka može da dobije novi par zečeva. Pretpostavimo da zečevi ne umiru i da ženka uvek daje novi par (muško-žensko) svaki mesec od drugog meseca. Koliko će parova biti na kraju prve godine.


Broj parova
1
1
2
3
5

algoritam

Fibonačijev metod



1. Odrediti interval L_0 [a,b] (a<b), koji sadrži,tačku x* i specificirati rezoluciju-tačnost aproksimacije ε>0. Interval
2. Odrediti najmanji prirodan broj *n* koji zadovoljava uslov:

$$F_n > \frac{1}{\varepsilon}(b-a) \quad \text{ili} \quad F_n > \frac{L_0}{\varepsilon}$$

3. Izračunati prvi interval (prva iteracija)

$$x_1 = a + \frac{F_{n-2}}{F_n}(b-a)$$

$$x_2 = a + b - x_1$$

4. Izračunati k ti interval i ponavljati postupak (korak 4.) sve do k=n

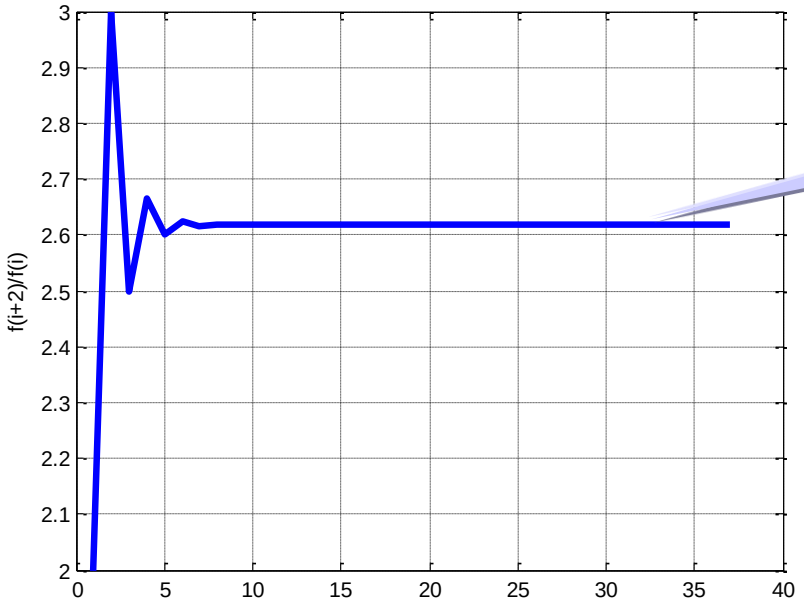
Odrediti $f(x_1)$ i $f(x_2)$, pa napraviti novi set tačaka a i b po principu:

Ako je f(x1) <= f(x2) tada eliminišemo x > x2 i b = x2, x2=x1, x1 = a + b - x1.
Ako f(x1) > f(x2) tada eliminišemo x < x1 i a = x1, x1=x2, x2 = a + b - x2
Ako f(x1) = f(x2) tada biramo novi par tačaka

```
function [x, fx] = fibonacijeva_metoda(fun, a, b, tol)

% Fibonacci-jev postupak minimizacije funckcije FUN jedne
promenljive.
% Funkcija mora biti unimodalna nad intervalom [a, b].
% Tol je trazena sirina intervala u kome se nalazi minimum.
% Minimum je u X i ima vrednost FX

%% Korak 1 - Trazimo najmanji broj n koji zadovoljava uslov
n = 1;
while tol < (b-a) / fibonacci_number(n)
    n = n + 1;
end

%% Korak 2 - Odredjujemo pocetne tacke
x1 = a + fibonacci_number(n-2) / fibonacci_number(n) * (b - a);
x2 = a + b - x1;

f1 = feval(fun, x1);
f2 = feval(fun, x2);
```

```
%
%% Korak 3 - Iteracije
% Radimo n-1 iteracija, posle cega je (b-a) < tol
for k = 2:n

    % Smanjujemo interval
    if f1 <= f2
        b = x2;
        x2 = x1; f2 = f1;
        x1 = a + b - x1; f1 = feval(fun, x1);
    else
        a = x1;
        x1 = x2; f1 = f2;
        x2 = a + b - x2; f2 = feval(fun, x2);
    end

    % Azuriramo resenje
    if f1 < f2
        x = x1; fx = f1;
    else
        x = x2; fx = f2;
    end
end
```

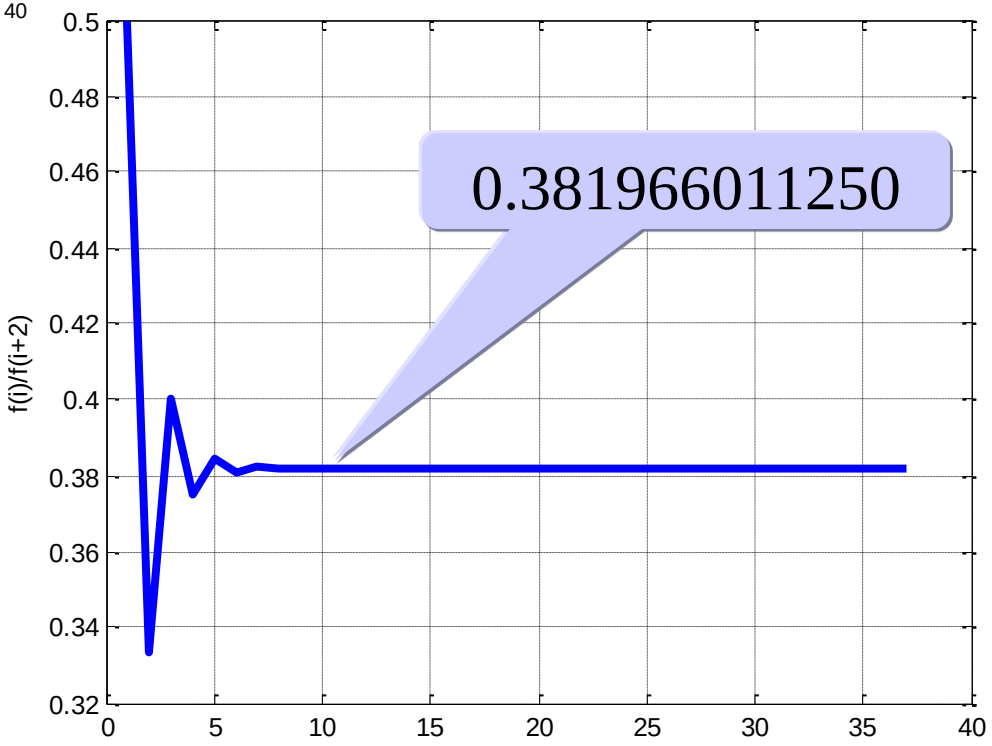
```
[x, fx] = fibonacijeva_metoda(@f, 0, 3, 0.0001)

x =
    1.3989
fx =
  -19.8016
```



2.618033988749

F(i)/F(i+2)

F(i+2)/F(i)

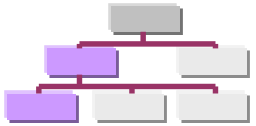


0.381966011250

Posle 15 brojeva podudaraju se na 5 decimalnih mesta

$$\frac{F_n}{F_{n+2}} \approx \frac{3-\sqrt{5}}{2}$$

Metod direktnog pretraživanja



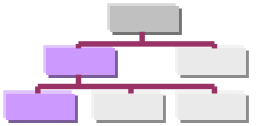
Metod Zlatnog Preseka

Jedan od nedostataka Fibonačijeve metode sastoji se u tome da se odredi broj iteracija *n* koji garantuje da tačka optimuma leži unutar željenog intervala.

Da bi to izbegli stavljamo da je odnos

$$\frac{F_n}{F_{n+2}} \approx c = \frac{3-\sqrt{5}}{2} = 0.38197$$

Tada algoritam postaje



1. Odrediti interval [a,b] (a<b), koji sadrži,tačku x* i specificirati rezoluciju-tačnost aproksimacije $\varepsilon>0$.
2. Izračunati

$$c = \frac{3-\sqrt{5}}{2} \approx 0.38197$$

3. Izračunati prvi interval

$$x_1 = a + c(b-a)$$

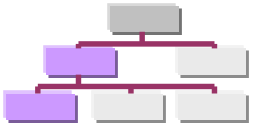
$$x_2 = a + b - x_1$$

4. Dok nije (b-a)< ε

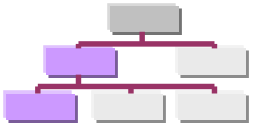
Odrediti $f(x_1)$ i $f(x_2)$, pa napraviti novi set tačaka a i b po principu:

Ako je f(x1) <= f(x2) tada eliminišemo x > x2 i b = x2, x2=x1, x1 = a + c*(b - a)
Ako f(x1) > f(x2) tada eliminišemo x < x1 i a = x1, x1 = x2, x2 = b - c*(b - a)
Ako f(x1) = f(x2) tada biramo novi par tačaka

Metod Zlatnog Preseka

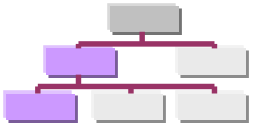


```
function [x, fx] = metoda_zlatnog_preseka(fun, a, b, tol)

% Zlatni presek - postupak minimizacije funckcije FUN jedne
promenljive.
% Funkcija mora biti unimodalna nad intervalom [a, b].
% Tol je trazena sirina intervala u kome se nalazi minimum.
% Minimum je u X i ima vrednost FX

%% Korak 1 - Odredjujemo pocetne tacke

% Odnos zlatnog preseka (celine prema manjem delu)
c = (3 - sqrt(5)) / 2;

% Racunamo pocetne tacke po zlatnom preseku
x1 = a + c * (b - a);
x2 = a + b - x1;

f1 = feval( fun, x1 );
f2 = feval( fun, x2 );
```



```
% %% Korak 2 - Iterativno smanjujemo interval dok ne dobijemo
zeljenu preciznost
while (b - a) > tol

    % Smanjujemo interval
    if f1 <= f2
        b = x2;
        x2 = x1; f2 = f1;
        x1 = a + c*(b - a); f1 = feval(fun, x1);
    else
        a = x1;
        x1 = x2; f1 = f2;
        x2 = b - c*(b - a); f2 = feval(fun, x2);
    end

    % Azuriramo resenje
    if f1 < f2
        x = x1; fx = f1;
    else
        x = x2; fx = f2;
    end
end
```

Metod Zlatnog Preseka



```
>> [x, fx] = metoda_zlatnog_preseka(@f, 1, 3, 0.0001)

x =

    1.3989


fx =

  -19.8016
```

Zašto se zove Zlatni Presek

$$\frac{AC}{BC} = \frac{BC}{AB}$$

Proveriti za domaći

Ako : AC=1 BC=x AB=1-x

Tada je: $\frac{1}{x} = \frac{x}{1-x}$

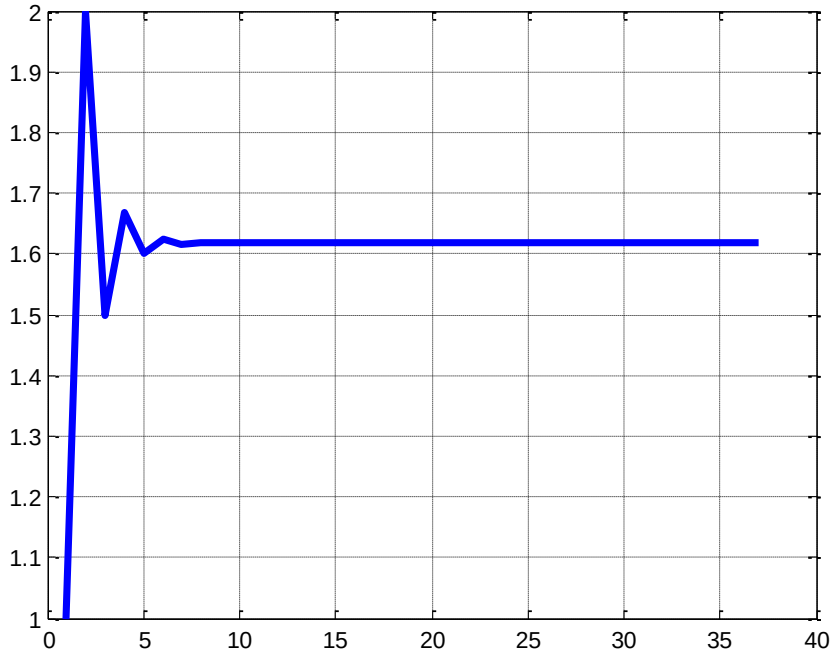
Rešiti kvadratnu jednačinu:

$$x^2 = 1 - x \quad 0 = x^2 + x - 1 \quad x = \frac{-1 \pm \sqrt{5}}{2}$$

Kako je x negativnu vrednost eliminišemo pa dobijamo

$$x = \frac{\sqrt{5}-1}{2} = 0.618034$$

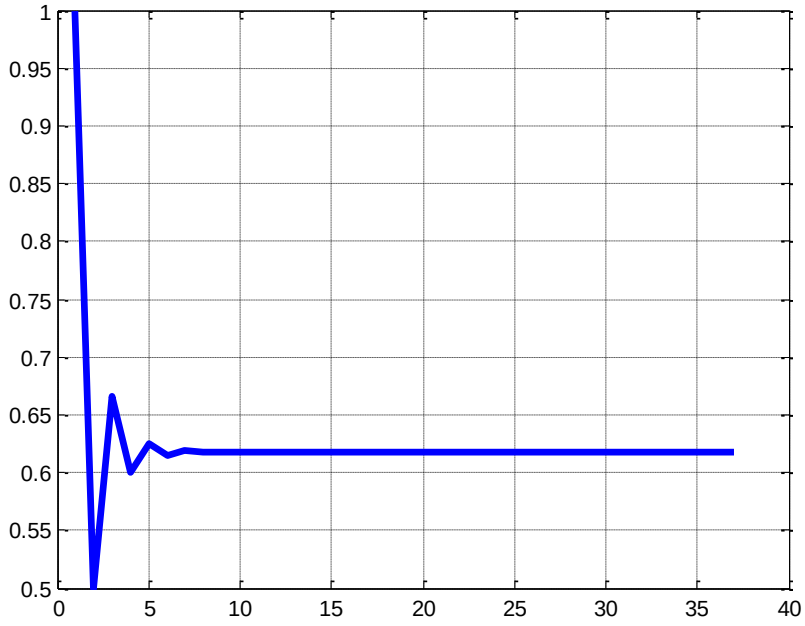
Metod Zlatnog Preseka



F(i+1)/F(i)

$$\frac{F_n}{F_{n+1}} \approx \frac{\sqrt{5}-1}{2}$$

F(i)/F(i+1)



Simbolički

- Phi - “Fi” - Φ

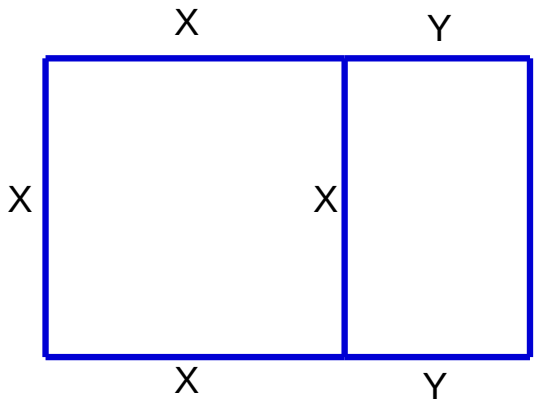
$$\Phi = \frac{\sqrt{5}+1}{2} = 1.6180...$$

- phi - “fi” - ϕ, φ

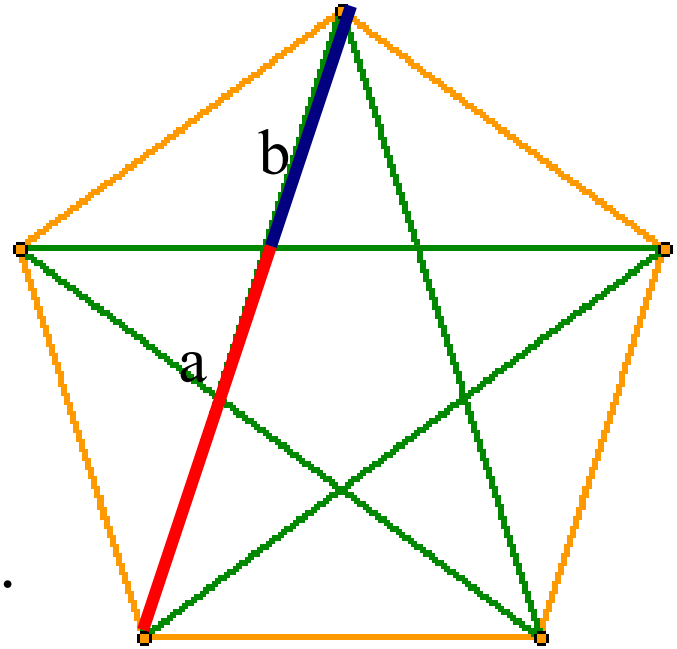
$$\phi = \frac{\sqrt{5}-1}{2} = 0.6180...$$

- Tau - τ

Geometrijski



$$\frac{x}{y} = \frac{x+y}{x}$$



$$\Phi = \frac{a}{b} = 1.6180...$$

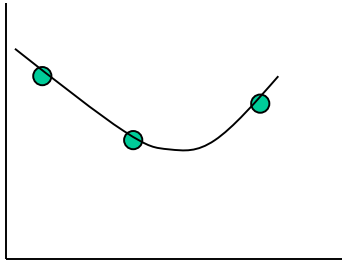
Metode aproksimacije polinomom

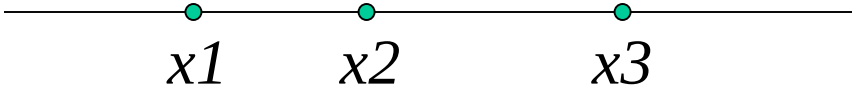
Osnovna ideja ovih metoda

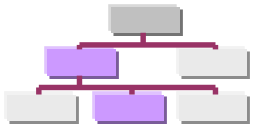
f-ja se aproksimira polinomom *y(x)* na intervalu I koji sadrži optimum, odredi se minimum *min y(x)=xopt*, u okolini *xopt* se formira novi interval (manji od prethodnog) i vrši se nova aproksimacija

Metod Parabole

$$y(x) = a + b\,x + c\,x^2$$

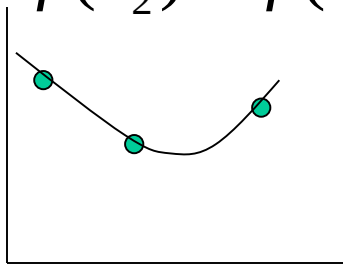






$$\mathbf{y(x) = a + b\,x + c\,x^2}$$

- traže se tri $x_1<x_2<x_3$ tačke tako da je $f(x_1) \geq f(x_2) \leq f(x_3)$ tada je i $x_1<x^*<x_3$



- Reši se sistem jednačina po *a*, *b*, *c*

  $a+bx_1+cx_1^2=f(x_1)$
  $a+bx_2+cx_2^2=f(x_2)$
  $a+bx_3+cx_3^2=f(x_3)$

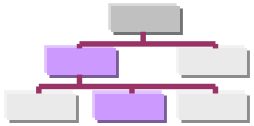
- Uslov minimuma parabole: $y'(x)=0$ da je

$$x_{opt} = -\frac{b}{2c}$$

- sada x_{opt} i dve susedne tačke od x_1, x_2, x_3 formiraju novu trojku i postupak se nastavlja. Uporediti x_{opt} i x_2 manja od njih dve je nova x_2 a tačke levo i desno čine x_1 i x_3.
- postupak se prekida kada je $\left| f(x_{opt}) - y(x_{opt}) \right| \leq \xi$

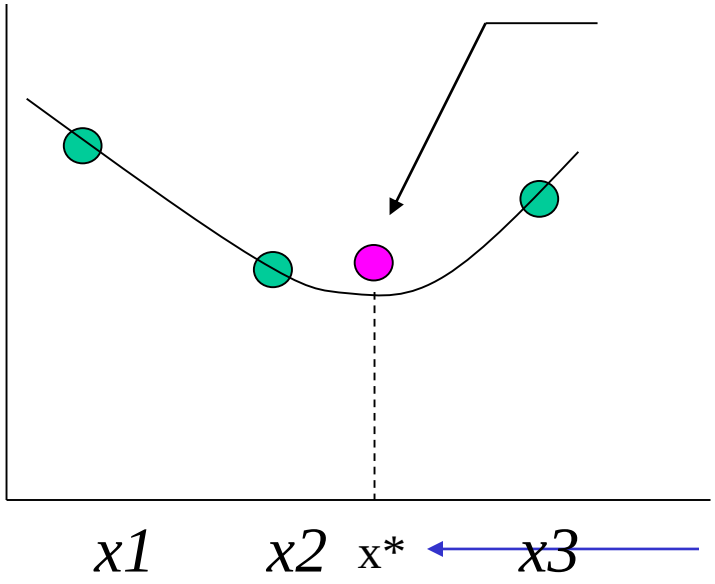
algoritam

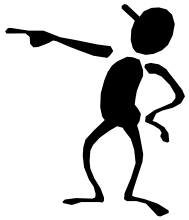
Metod Parabole



$$y(x) = a + b\,x + c\,x^2$$

$$f(x_1) \geq f(x_2) \leq f(x_3)$$





Optimum aproksimacije

$x_2 = x_2$

$x_1 = x_1$

$x_3 = x^*$

$$x^* = \frac{1}{2}\frac{(x_2^2 - x_3^2)f_1 + (x_3^2 - x_1^2)f_2 + (x_1^2 - x_2^2)f_3}{(x_2 - x_3)f_1 + (x_3 - x_1)f_2 + (x_1 - x_2)f_3}$$

```
function [x, fx, n] = parabola( fun, x1, x3, tol )

X = [ x1 (x1+x3)/2 x3 ]';
Y = [ [1 1 1]' X X.*X ];
F = feval(fun, X);
abc = Y \ F;
x = -abc(2) / 2 / abc(3);
fx = feval(fun,x);
n = 0;

while abs( [1 x x^2] * abc - fx ) > tol

  if (x > X(2)) & (x < X(3))
    if (fx < F(2)) & (fx < F(3))
      X = [X(2); x; X(3)];  F = [F(2); fx; F(3)];
    elseif (fx > F(2)) & (fx < F(3))
      X = [X(1); X(2); x];  F = [F(1); F(2); fx];
    else
      error('Greska: Fopt > min(F2,F3)')
    end                                          %...
```

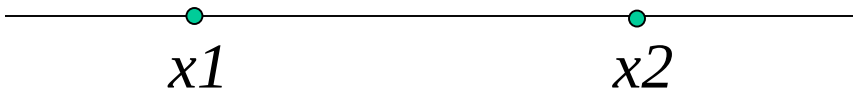
```
  % ...
  elseif (x > X(1)) & (x < X(2))
    if (fx < F(1)) & (fx < F(2))
      X = [X(1); x; X(2)];  F = [F(1); fx; F(2)];
    elseif (fx > F(2)) & (fx < F(1))
      X = [x; X(2); X(3)];  F = [fx; F(2); F(3)];
    else
      error('Greska: Fopt > min(F1,F2)')
    end
  else
      error('x lezi van granica')
  end

  Y = [ [1 1 1]' X X.*X ];
  abc = Y \ F;
  x = -abc(2) / 2 / abc(3);
  fx = feval(fun, x);
  n = n + 1;
end
```

```
> [x,fx,n,rez] = parabola( 'fun', 0, 2, 0.0001 )
x =
   1.39893632158021

fx =

 -19.80161279629130

n =
     3
rez =
  Columns 1 through 5

                0                0   1.00000000000000   2.00000000000000                0
   1.00000000000000   1.00000000000000   1.40000000000000   2.00000000000000 -18.00000000000000
   2.00000000000000   1.00000000000000   1.40000000000000   1.40774907749077 -18.00000000000000
   3.00000000000000   1.00000000000000   1.39896822541732   1.40000000000000 -18.00000000000000
  Columns 6 through 10
 -18.00000000000000 -16.00000000000000                0 -28.00000000000000  10.00000000000000
 -19.80159999999999 -16.00000000000000   1.67999999999994 -30.51999999999992  10.83999999999997
 -19.80159999999999 -19.80073936314690   2.34974619081601 -31.66813632711318  11.31839013629717
 -19.80161279629130 -19.80159999999999   2.35228740652486 -31.67249269689978  11.32020529037492
  Columns 11 through 12
   1.40000000000000   0.20160000000000
   1.40774907749077   0.00065092250923
   1.39896822541732   0.00087268594479
   1.39893632158021   0.00000001152232
```

Metode aproksimacije polinomom

Kubna Metoda

$$y(x)=a+bx+cx^2+dx^3$$



$$f'(x_1)<0 \quad f'(x_2)>0 \quad x_1<x_2$$

algoritam

- aproksimira *f(x)* polinomom 3. reda
  $y(x)=a+bx+cx^2+dx^3$ $f'(x_1)<0 \quad f'(x_2)>0 \quad x_1<x_2$
- Koeficijenti se mogu odrediti na 2 načina:
  - poznavanjem *f(x)* u 4 tačke ili
  - poznavanjem *f(x)* i *f'(x)* u 2 tačke.

    $a+bx_i+cx_i^2+dx_i^3=f(x_i)$, i=1,2

    $b+2cx_i+3dx_i^2=f'(x_i)$, i=1,2
- Kod rešavanja *y'(x)* =0 dobijaju se dva rešenja, a uzima se ono koje leži u intervalu $[x_1,x_2]$ i ima manju vrednost (za minimum)
  Ako je $f'(x_{opt})<0$, $x_1=x_{opt}$
  inače $x_2=x_{opt}$

- postupak se prekida kada je $\left|f(x_{opt})-y(x_{opt})\right|\le\xi$

```
function [x,fx,n]=kubna(fun,dfun,x1,x2,tol)
X=[x1; x2];
Y=[[1; 1] X X.^2 X.^3
   [0; 0] [1; 1] 2*X  3*X.^2];
F=[feval(fun,X); feval(dfun,X)];
abcd=Y\F; b=abcd(2); c=abcd(3); d=abcd(4);
D=sqrt(4*c*c-12*b*d);
xa=(-2*c-D)/6/d;
xb=(-2*c+D)/6/d;
if feval(fun,xa) < feval(fun,xb)
  x=xa;
else
  x=xb;
end
fx=feval(fun,x);
n=0;
```

```
while abs([1 x x^2 x^3]*abcd - fx))>tol
  if feval(fun,xa) < feval(fun,xb)
    X(2)=x; else X(1)=x; end
  Y=[[1; 1] X X.^2 X.^3
     [0; 0] [1; 1] 2*X  3*X.^2];
  F=[feval(fun,X); feval(dfun,X)];
  abcd=Y\F; b=abcd(2); c=abcd(3); d=abcd(4);
  D=sqrt(4*c*c-12*b*d);
  xa=(-2*c-D)/6/d;
  xb=(-2*c+D)/6/d;
  if feval(fun,xa) < feval(fun,xb)
    x=xa;
  else
    x=xb;
  end
  n=n+1;
  fx=feval(fun,x);
end
```

```
>> [x,fx,n,rez] = kubna( 'fun', 'dfun1', 0, 2, 0.00001 )
x =
   1.39893257541003

fx =
 -19.80161281065906

n =
     8
rez =
  Columns 1 through 5
   1.00000000000000                0   2.00000000000000  12.00000000000000  20.78460969082653
   2.00000000000000                0   1.46410161513775  17.93335800641838  18.88286623804345
   3.00000000000000                0   1.40838201847866  18.38965402090047  18.52766984613476
   4.00000000000000                0   1.40032920221022  18.45340543137662  18.47381331793818
   5.00000000000000                0   1.39913947389076  18.46277783625626  18.46580251318223
   6.00000000000000                0   1.39896316507386  18.46416574522066  18.46461418913156
   7.00000000000000                0   1.39893702570757  18.46437149298518  18.46443798342407
   8.00000000000000                0   1.39893315005427  18.46440199847762  18.46441185703840
  Column 6
   1.46410161513775
   1.40838201847866
   1.40032920221022
   1.39913947389076
   1.39896316507386
   1.39893702570757
   1.39893315005427
   1.39893257541003
```

Osobine kubne metode

- optimum uvek leži u [*x1,x2*]
- brža je od metode parabole, ali zahteva više računarskih operacija (stepen konvergencije je superlinearan)
- Minimum *y(x)* na intervalu [*x1,x2*] se može izračunati direktno

$$x^* = x_2 - \frac{f_2' + w - z}{f_2' - f_1' + 2w}(x_2 - x_1)$$

$$z = 3\frac{f_1 - f_2}{x_2 - x_1} + f_1' + f_2'$$

$$w = \sqrt{z^2 - f_1' \cdot f_2'}$$