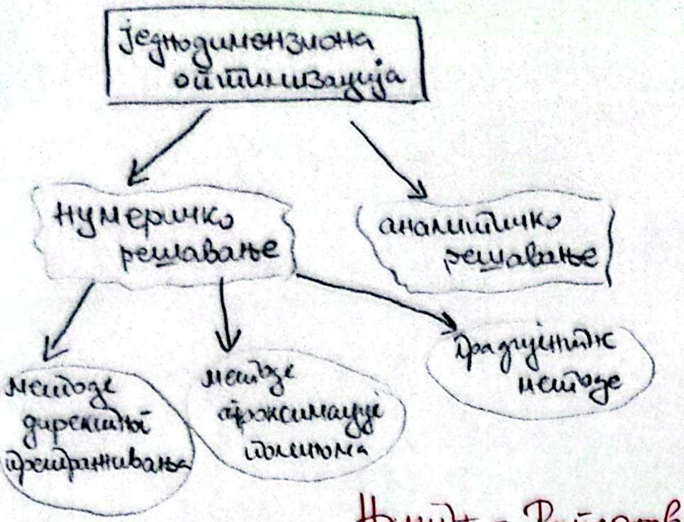
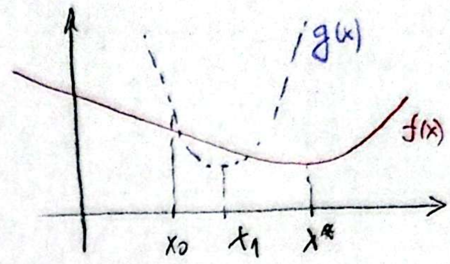
ЈЕДНОДИМЕНЗИОНЕ НУМЕРИЧКЕ МЕТОДЕ



- Градијентне методе - основна идеја је наћи стационарне тачке функције $f'(x)=0$ (ако је ф-ја диференцијабилна до реда који нам је потребан)



Њутн - Рафсонов метод

$f(x)$ се развија у Тејлоров ред око x_0

$g(x)$ је парабола чији је екстрем у x_1, а узима се као полазна тачка за следећу итерацију

$$g'(x_1)=0$$

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

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

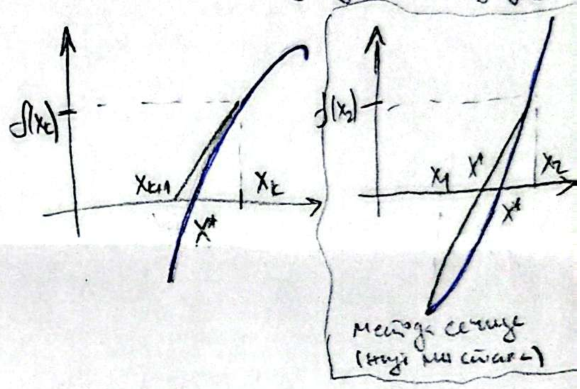
$$x_1-x_0=\frac{-f'(x_0)}{f''(x_0)}$$

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

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

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

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



- почетно погађање: $x_1=x_0-\frac{f'(x_0)}{f''(x_0)}$
- итеративни поступак: $x_{n+1}=x_n-\frac{f'(x_n)}{f''(x_n)}$
- критеријум заустављања: $\varepsilon_{n+1}=\left|\frac{x_{n+1}-x_n}{x_{n+1}}\right|$ $\quad |x_n-x_{n-1}|<\varepsilon \rightarrow$ нпр. ћемо се зауставити када вредности итерација постану веома блиске

- конвергенција ЗАВИСИ ОД ДОБРОГ ПОЧЕТНОГ ПОГАЂАЊА

x_{n+1} - вредност наредне итерације
x_n - тренутна вредност
$f'(x_n)$ - први извод ф-је (градијент)
$f''(x_n)$ - други извод у тачки x_n

ε - толеранција ф-је

- присилан крај - да се унапред одреди макс број итерација јер ако се не одабере добро почетно погађање, алгоритам може да дивергира и проводити критеријум никад неће бити задовољен

Метод сечице

- основна разлика ове методе и Њутн-Рафсона је то што се овде уводи апроксимација за други извод

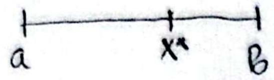
$$\boxed{f''(x_n)=\frac{f'(x_n)-f'(x_{n-1})}{x_n-x_{n-1}}}$$

итеративна формула: $\boxed{x_{n+1}=x_n-f'(x_n)\frac{x_n-x_{n-1}}{f'(x_n)-f'(x_{n-1})}}$

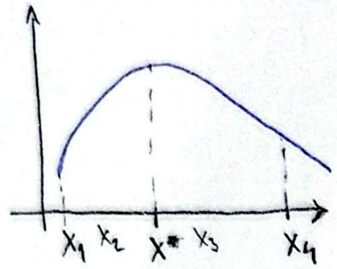
на исти фазон

- није потребно да ф-ја буде диференцијабилна до другог извода, али су овде неопходне две почетне тачке. Одабиром доброг почетног интервала омогућавамо алгоритму да конвергира ка оптимуму. Овај алгоритам ће смањити брзину конвергенције у поређењу са Њутн-Рафсоном

- Методе директног претраживања основне су методе једнодимензионе оптимизације сматрају их „кичмом" нелинеарних оптимизационих алгоритама и своде се на ПРЕТРАЖ- ЗАТВОРЕНОГ ИНТЕРВАЛА; често претпостављамо да је фја УНИМОДАЛНА

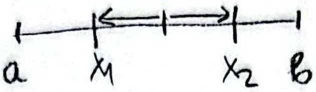


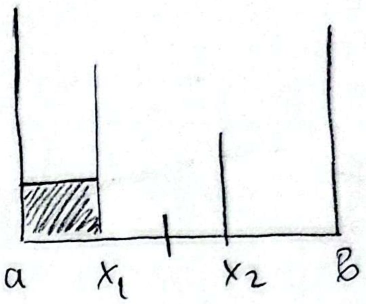
детаљно претраживање захтева $N = \frac{b-a}{\varepsilon} + 1$ прорачуна у интервалу
где је ε - резолуција



$y(x_1) < y(x_2) < y(x^*)$ ако је $x_1 < x_2 < x^*$

$y(x^*) > y(x_3) > y(x_4)$ ако је $x^* < x_3 < x_4$





процедура да би се нашао min $f(x)$:

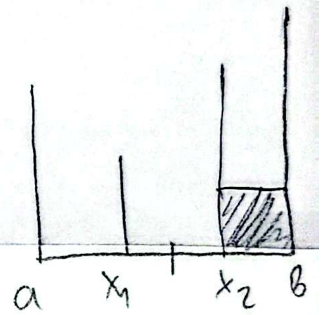
0) претпоставимо интервал $[a,b]$

1) нађемо $x_1 = a + \frac{b-a}{2} - \frac{\varepsilon}{2}$ $\quad x_2 = a + \frac{b-a}{2} + \frac{\varepsilon}{2}$

2) поредимо $f(x_1)$ и $f(x_2)$

3) ако је $f(x_1) < f(x_2)$, елиминишемо $x > x_2$ и $b = x_2$
ако је $f(x_1) > f(x_2)$, елиминишемо $x < x_1$ и $a = x_1$
ако су једнаки, бирамо нови пар тачака

4) настављамо док не буде $< 2\varepsilon$



- функција је унимодална ако има један локални екстремум на одређеном интервалу који је уједно и глобални екстремум на том интервалу

Фибоначијев метод

1. одр. интервал $L_0[a,b]$ $(a<b)$ који садржи x^* и специфицирати резолуцију тачности $\varepsilon>0$
2. одр. најмањи пр. бр. n који задовољава

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

3. изр. први интервал:

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

$$x_2 = a+b-x_1$$

4. изр. k-ти интервал и понављати поступак

ИСТО

$f(x_1) <= f(x_2)$ елиминишемо $x>x_2$ и $b=x_2$, $x_2=x_1$, $x_1=a+b-x_1$

за $f(x_1)>f(x_2)$ елим. $x<x_1$ и $a=x_1$, $x_1=x_2$, $x_2=a+b-x_2$

за $f(x_1)=f(x_2)$ нови пар тачака

Метод златног пресека

$$\frac{F_{n-2}}{F_n} \approx \frac{3-\sqrt{5}}{2} = 0.38196$$

$b-a<\varepsilon$ - критеријум заустављања

1. одр. интервал $L_0[a,b]$ $(a<b)$ који садржи x^* и специфицирати резолуцију тачности $\varepsilon>0$
2. израчунати $c=\frac{3-\sqrt{5}}{2}$
3. изр. први интервал:

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

$$x_2 = a+b-x_1$$

4. док није $(b-a)<\varepsilon$:

$f(x_1)<=f(x_2)$ елиминишемо $x>x_2$ и $b=x_2$, $x_2=x_1$, $x_1=a+c*(b-a)$

$f(x_1)>f(x_2)$ елиминишемо $x<x_1$ и $a=x_1$, $x_1=x_2$, $x_2=b-c*(b-a)$

$f(x_1)=f(x_2)$ нови пар тачака

ВИШЕДИМЕНЗИОНЕ НУМЕРИЧКЕ МЕТОДЕ

Метод најбржег пада

- градијент - вектор који показује смер и брзину највеће промене фје
- рецимо ако имамо фју са више променљивих, градијент те фје ∇f је вектор који садржи ПАРЦИЈАЛНЕ ИЗВОДЕ ФУНКЦИЈЕ ПО СВАКОЈ ПРОМЕНЉИВОЈ

$$\nabla f(x,y,z) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}\right)$$

промена фје f када мењамо само x (y и z су конст.)

- вектор градијента показује смер највеће промене вредности фје

- овим методом тражи се минимум диференцијабилне фје $f(x)=f(x_1,x_2,\dots,x_n)$ тако што се у свакој итерацији решење помера у правцу негативног градијента од f

$$\nabla f = \left[\frac{\partial f}{\partial x_1}\ \frac{\partial f}{\partial x_2} \cdots \frac{\partial f}{\partial x_n}\right]^T$$

1. иницијални задатак: одабере се почетно решење x_0, корак $\gamma>0$, толеранција $\varepsilon>0$, макс. број корака N
2. тело метода које примењујемо итеративно: $\boxed{x_{k+1} = x_k - \gamma \nabla f(x_k)}$
3. критеријум заустављања - на крају сваке итерације проверавамо $\|\nabla f(x_k)\| \le \varepsilon$

Метод најбржег пада са моментумом

$\boxed{v_k = w v_{k-1} + \gamma \nabla f(x_k)}$ $\boxed{x_{k+1} = x_k - v_k}$

- даје моменат одн. СМЕР ПРЕТХОДНОГ ГРАДИЈЕНТА у израчунавању како би се осцилације смањиле

  v - вектор брзине који садржи инфо. о претходном правцу кретања

  w - коеф. момента (колико памтимо претх. правац) - типично између 0 и 1

- ради тако што чува правац претх. корака да би стабилизовао путању у правцу мин.
- стабилнија конвергенција и смањење осцилација у близини минимума
- међутим, подешавање параметара може бити компликовано

- основни проблем са свим до сада размотреним алгоритмима је што је брзина адаптације иста по свим осама, што значи да је тешко подесити параметре када се при прорачуну знатно брже мења по једној оси него по другој
- ако МАЛА ПРОМЕНА неке променљиве доводи до ВЕЛИКИХ ПРОМЕНА к.о. – променљиву мењати полако
- ако МАЛА ПРОМЕНА неке променљиве доводи до НЕЗНАТНИХ ПРОМЕНА к.о. – променљиву мењати брзо са већим корацима

дакле брзина адаптације треба да буде иста за сваку осу

Адаград (адаптивни градијент)

$$\boxed{X_{k+1,i} = X_{k,i} - \frac{\eta}{\sqrt{G_{k,i} + \varepsilon_1}} g_{k,i}}$$

g - градијент ф-је у претходној итерацији (смер промене за сваку променљиву)

G - сума квадрата свих претходних градијената

η - почетна учесталост учења, скалира се за сваку променљиву у зависности од G

ε_1 - мали параметар служи за регулацију израза у почетним итерацијама (да не делимо) (10^{-8})

- основни проблем јесте акумулација градијената у величини G (која се константно током времена повећава). Ово доводи до ефективног смањења дужине свих корака по свим димензијама, а тиме се ефикасност алгоритма из итерације у итерацију смањује.
- резултат што РАЧУНА СУМУ КВАДРАТА ПРЕТХОДНИХ ГРАДИЈЕНАТА како би прилагодио корак у свакој итерацији

Адам (адаптивно процењивање момента)

- један је од најчешће коришћених савремених модификација алгоритма најстрмијег пада

$$m_k = w_1 m_{k-1} + (1-w_1) g_k \qquad \hat{m}_k = \frac{m_k}{1-w_1}$$

$$v_k = w_2 v_{k-1} + (1-w_2) g_k^2 \qquad \hat{v}_k = \frac{v_k}{1-w_2}$$

$$\boxed{X_{k+1} = X_k - \frac{\eta}{\sqrt{\hat{v}_k + \varepsilon_1}} \hat{m}_k}$$

g - градијент ф-је у претходној итерацији, смер у којем се врши оптимизација за сваку променљиву

m - ПРОСЕЧНИ ГРАДИЈЕНТ (први момент) до претходне итерације, што помаже да се ухвати дугорочни смер кретања у оптимизацији

v - ПРОСЕЧАН КВАДРАТ ГРАДИЈЕНТА (други момент) што помаже да се контролише величина корака и ублаже нагле промене у градијентима

w_1, w_2 - коефицијенти за експоненцијално избледњавање првог и другог момента (обично вредности 0,9 и 0,999)

ε_1 - мали број да спречи дељење нулом (обично 10^{-8})

η - почетна учесталост учења скалирана са $\hat{v}$ како би корак био стабилнији и прилагођавао различитим променљивим