ԳԼՈՒԽ 2

Ավտոմատ դիֆերենցում : Ավտոմատ դիֆերենցման մեթոդները: Դուալ թվերի կապը դիֆերենցման հետ:

2.1 Դուալ թվեր

Դուալ թվերը առաջին անգամ սահմանել և նկարագրել է անգլիացի մաթեմատիկոս Վիլյամ Քլիֆորդը 1873 թվականին : Դուալ թվերը սովորաբար ներկայացվում է հետևյալ բանաձևով.

$$X = a + b\xi$$

որտեղ a-ն և b-ն իրական թվեր են, իսկ ξ-ն այնպիսի աբստրակտ պարամետր է (միավոր է), որն ունի հետևյալ հատկությունը.

$$\xi^2 = 0$$

Դուալ թիվը հավասար է 0-ի, երբ $a = 0$, և $b = 0$: a-ն կոչվում է դուալ թվի գլխավոր մաս (նշանակվում է Re-ով), իսկ $b\xi$-ն կոչվում է դուալ թվի դուալ կամ աբստրակտ մաս (նշանակվում է Im-ով): Ընդ որում

$$\xi \neq 0$$

$$\xi^k = 0 \ , \quad k > 1$$

Ենթադրենք ունենք հետևյալ դուալ թվերը.

$$X = a + b\xi \ , \quad Y = c + d\xi$$

Դուալ թվերի համար սահմանենք գումարման, հանման, բազմապատկման և բաժանման բանաձևերը.

$$X + Y = (a + b\xi) + (c + d\xi) = (a + c) + \xi\,(b + d)$$

$$X - Y = (a + b\xi) - (c + d\xi) = (a - c) + \xi\,(b - d)$$

$$X * Y = (a + b\xi) * (c + d\xi) = (a * c) + \xi\,(a * d + c * b)$$

$$\frac{X}{Y} = \frac{(a + b\xi)}{(c + d\xi)} = \frac{a}{c} + \xi\frac{(c * b - a * d)}{c^2}, \quad c \neq 0$$

2.2 Դուալ թվերի կապը դիֆերենցման հետ

Այս ենթագլխում ցույց տանք թե ինչպես կարելի է ստանալ ֆունկցիայի մասնակի ածանցյալի արժեքը, և ֆունկցիայի արժեքը , ինչ-որ կետերում, օգտվելով դուալ թվերից: Հաշվենք դուալ թվից կախված ֆունկցիան, օգտվելով Թեյլորի շարքից.

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k$$

x-ի փոխարեն տեղադրենք $a + b\xi$, կստանանք.

$$f(a + b\xi) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(a + b\xi - a)^k = \sum_{k=0}^{\infty} \frac{f^{(k)}(a) * b^k * \xi^k}{k!} =$$

$$= f(a) + bf'(a)\xi + \sum_{k=2}^{\infty} \frac{f^{(k)}(a) * b^k * \xi^k}{k!} =$$

$$= f(a) + bf'(a)\xi + \xi^2 \sum_{k=2}^{\infty} \frac{f^{(k)}(a) * b^k * \xi^{k-2}}{k!} =$$

$$= f(a) + bf'(a)\xi$$

Այստեղից հետևում է որ, եթե $x = a$-ի փոխարեն տեղադրենք $f(x)$ ֆունկցիայում $x = a + b\xi$ արժեքը, ապա կստանանք և՛ ֆունկցիայի արժեքը $x = a$ կետում, այսինքն $f(a)$-ն, և՛ $f(x)$ ֆունկցիայի ածանցյալի արժեքը $x = a$ կետում, այսինքն $f'(a)$-ն: Դիտարկենք հետևյալ օրինակը.

<u>Օրինակ:</u>

Ենթադրենք տրված է հետևյալ ֆունկցիան՝ $f(x) = x^3$: Պահանջվում է գտնել ֆունկցիայի ածանցյալի արժեքը $x = 10$ կետում: Ինչպես գիտենք ֆունկցիայի ածացյալը տվյալ կետում հավասար է ՝ $f'(a) = 3 * 10^2 = 300$, իսկ ֆունկցիայի արժեքը տվյալ կետում ՝ $f(a) = 1000$: Այժմ հաշվենք ֆունկցիայի ածանցյալի արժեքը, օգտվլով դուալ թվերից: Տեղադրենք ֆունկցիայում $x = 10$ -ի փոխարեն $x = 10 + 1\xi$, կստանանք.

$$f(10 + 1\xi) = (10 + 1\xi)^3 = 1000 + 300\xi + 30\xi^2 + \xi^3$$

ըստ սահմանման $30\xi^2 = 0$, $\xi^3 = 0$, հետևում է որ $f(10 + 1\xi) = 1000 + 300\xi$, որտեղ 1000-ը մեր a-ն է, այսինքն ֆունկցիայի արժեքը տվյալ կետում, իսկ 300-ը $1 * b$-ն է, այսինքն ֆունկցիայի ածանցյալի արժեքը տվյալ կետում:

Այսպիսով ստավեց, որ դուալ թվերի միջոցով կարող ենք առանց ֆունկցիայի ածանցյալ հաշվելու, գտնել ֆունկցիայի ածանցյալի արժեքը տվյալ կետում: Դուալ թվերի այս հատկությունը կիրառվում է ավտոմատ դիֆերենցման ուղիղ և հակադարձ մեթոդների մեջ:

2.3 Ավտոմատ դիֆերենցում

Մաթեմատիկայում և համակարգչային հանրահաշվում ավտոմատ դիֆերենցումը իրենից ներկայացնում է մեթոդների հավաքածու, որոնք նախատեսված են համակարգչային ծրագրերի միջոցով հաշվել ֆունկցիայի ածանցյալիների թվային արժեքները: Ավտոմատ դիֆերենցմանը հաճախ անվանում են նաև ալգորիթմական կամ հաշվողական դիֆերենցում: Ացտոմատ դիֆերենցումը հիմնվում է այն փաստի վրա, որ ցանկացած բարդ համակարգչային ծրագիր, վերջ ի վերջո կառուցված է, և կազմված է տարրական թվաբանական գործողությունների (գումարում, հանում, բազմապատկում, բաժանում և այլն), և տարրական մաթեմատիկական ֆունկցիաների (օրինակ՝ $exp, log, sin, cos,$ և այլն) հաջորդականությունից: Այսիքն կամայական կարգի ածանցյալների հաշվարկը, իրենից ներկայացնում է այս գործողությունների հաջորդական կիրառումը:

Այս մեթոդը տարբերվում է ինչպես սիմվոլային դիֆերենցումից, այնպես էլ թվային դիֆերենցումից: Սիմվոլային դիֆերենցման ժամանակ երբեմն ստացվում է ծավալուն, անարդյունավետ արտահայտություններ, որը համակարգչային ծրագրով իրականացման ժամանակ առաջացնում է բարդություններ: Թվային դիֆերենցումը ևս ունի թերություններ, որոնց հիմնական պատճառը սխալի կլորացումն է: Այս երկու

դասական մոտեցումներն ունեն նաև խնդիրներ՝ բազմաթիվ փոփոխականներով ֆունցիաների մասնակի ածանցյալների հաշվարկման ժամանակ, ինչը շատ կարևոր է օպտիմալացման խնդիրներում, գրադիենտային մեթոդներում, որոնք հիմնականում կիրառվում են մեքենայական ուսուցման մեջ: Ավտոմատ դիֆերենցումը լուծում է նշված բոլոր խնդիրները: Ավտոմատ դիֆերենցումը կազմված է երկու հիմնական մեթոդից՝ ուղիղ մեթոդ և հակադարձ մեթոդ:

Դիտարկենք ավտոմատ դիֆերենցման մի օրինակ:

Օրինակ:

Ենթադրենք տրված է հետևյալ ֆունկցիան՝ $f(x) = 3x + 2$, և գտնենք $f(4)$-ը և $f'(4)$: Առաջին քայլով, $x = 4$ –ի փոխարեն տեղադրում ենք ֆունկցիայում $x = 4 + 1\xi$ արժեքը.

$$f(4 + 1\xi) = 3(4 + 1\xi) + 2 = 14 + 3\xi$$

որտեղ 14-ը ֆունկցիայի արժեքն է $x = 4$ կետում, իսկ 3ξ դուալ մասն է, որը համապատասխանում է $f'(4)$-ին: Այս մոտեցումը հայտնի է որպես ավտոմատ դիֆերենցման ուղիղ մեթոդ, որի ժամանակ ածանցյալը հաշվարկվում է զուգահեռաբար, հիմանկան ֆունկցիայի արժեքի հաշվարկման հետ, սկսած ալգորիթմի սկզբից մինչև վերջ:

2.4 ՈՒղիղ և Հակադարձ մեթոդ

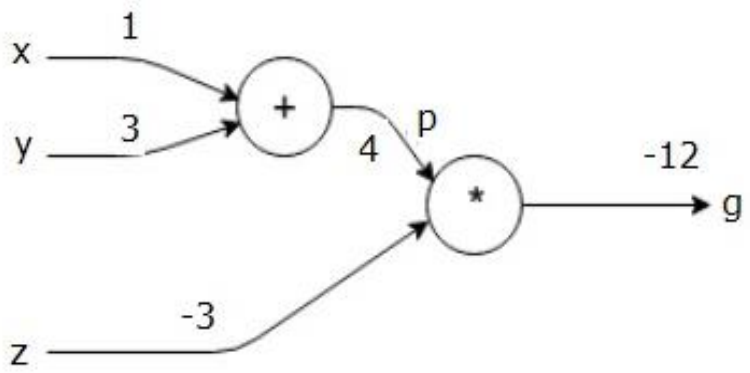
Սահմանում:

Հաշվողական գրաֆը այն գրաֆն է, որտեղ գագաթները (հանգույցները) իրենցից ներկայացնում են մաթեմատիկական գործողություններ (օրինակ՝ գումարում, բազմապատկում, և այլն), իսկ դրանց միջև եղած եզրերը (կողերը) իրենցից ներկայացնում են տվյալներ (փոփոխականներ կամ միջանկյալ արդյունքներ):

Հաշվողական գրաֆները օգտագործվում են հաշվարկները ներկայացնելու համար, որպես գործողությունների հաջորդականություն, որոնք պետք է կատարվեն: Այս պարագայում կարող ենք նմանեցնել այն ալգորիթմի բլոկ սխեմանների հետ:

Օրինակ:

Ենթադրենք տրված է հետևյալ ֆունկցիան՝ $f(x) = (x + y) * z$: Ներկայացնենք այն հաշվողական գրաֆի միջոցով.



նկար 2.1

Ինչպես տեսնում ենք մուտքային արժեքները երեքն են՝ x, y, և z : Գրաֆի գագաթները իրենցից ներկայացնում են գումարում և բազմապատկում, իսկ գրաֆի կողերը իրենցից ներկայացնում են սկզբնական , միջանկյալ, և վերջնական տվյալները:

Ավտոմատ դիֆերենցման հիմնական գաղափարը ինչպես նշվեց ֆունկցիայի տարալուծումն (վերլուծումն) է ավելի պարզ թվաբանական գործողությունների, և տարրական ֆունկցիաների: Այդ բոլոր գործողությունները ավելի նպատակահարմար է ներկայացնել հաշվողական գրաֆների միջոցով: Այդ նույն մոտեցումը ընկած է համակարգչային ծրագրի իրականացման հիմքում:

Հաշվողական գրաֆի միջոցով ուղիղ և հակադարձ մեթոդների տարբերությունը հասկանալու համար, դիտարկենք մի օրինակ:

Օրինակ:

Ենթադրենք տրված է երկու փոփոխականներից կազմված հետևյալ ֆունկցիան՝ $f(x,y)$:

$$f(x,y) = (x+y)^2 + 2xy$$

Պետք է գտնել $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$ -ը $(2;-4)$ կետում : Կատարենք նշանակում $a = (x+y)$, $b = 2xy$:

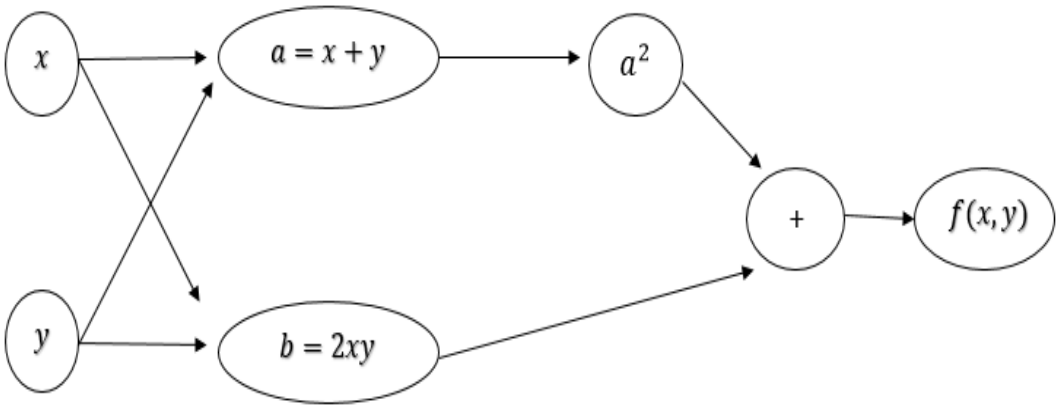
$$f(x,y) = a^2 + b$$

Հաշվենք ֆունկցիայի ածանցյալը a, և b պարամետրերի ներմուծման դեպքում.

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial a} * \frac{\partial a}{\partial x} + \frac{\partial f}{\partial b} * \frac{\partial b}{\partial x}$$

$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial a} * \frac{\partial a}{\partial y} + \frac{\partial f}{\partial b} * \frac{\partial b}{\partial y} \qquad (1)$$

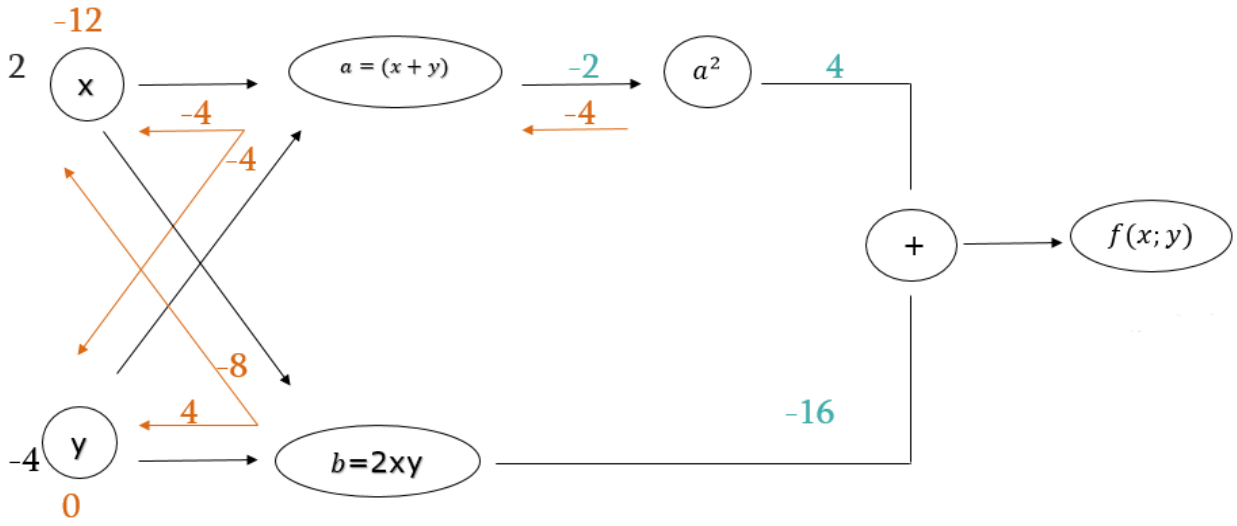
Ներկայացնենք այն հաշվողական գրաֆի միջոցով.



նկար 2.2

<u>Հակադարձ մեթոդ:</u>

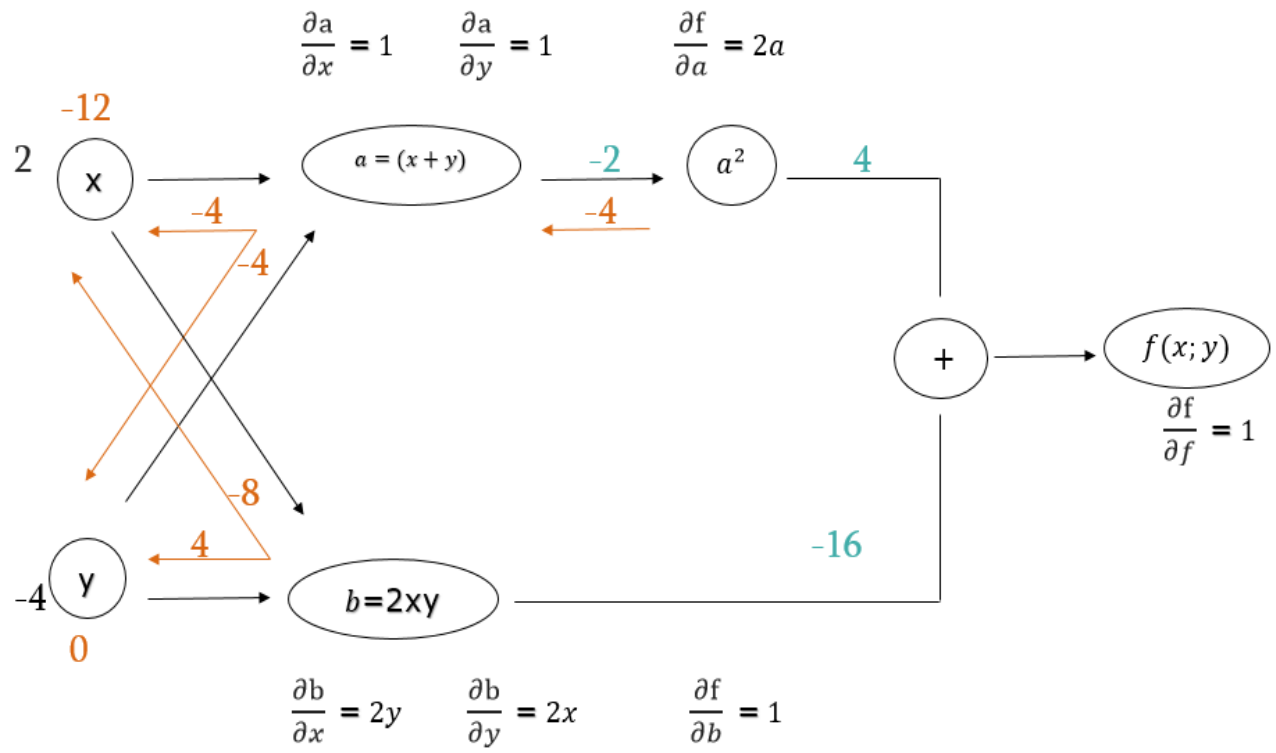
Առաջին քայլով տեղադրում ենք x-ի և y-ի արժեքները (նկ. 2.3) համապատասխան գագաթում (հանգույցում), և ստանում ենք ելքային արժեքները (նշված է կանաչ գույնով): Այնուհետև հաշվում ենք բոլոր մասնակի ածանցյալները, որոնք մասնակցում են (1) – ում, և տեղադրում ենք x-ի և y-ի արժեքները (նշված է նարնջագույնով):



նկար 2.3

Արդյունքում, x-ի բոլոր մուտքային արժեքների գումարը, և y-ի բոլոր մուտքային արժեքների գումարը կհանդիսանան համապատասխանաբար $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$ -ը $(2;-4)$ կետում (նկ. 2.4):

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial a} * \frac{\partial a}{\partial x} + \frac{\partial f}{\partial b} * \frac{\partial b}{\partial x}, \qquad \frac{\partial f}{\partial y} = \frac{\partial f}{\partial a} * \frac{\partial a}{\partial y} + \frac{\partial f}{\partial b} * \frac{\partial b}{\partial y}$$

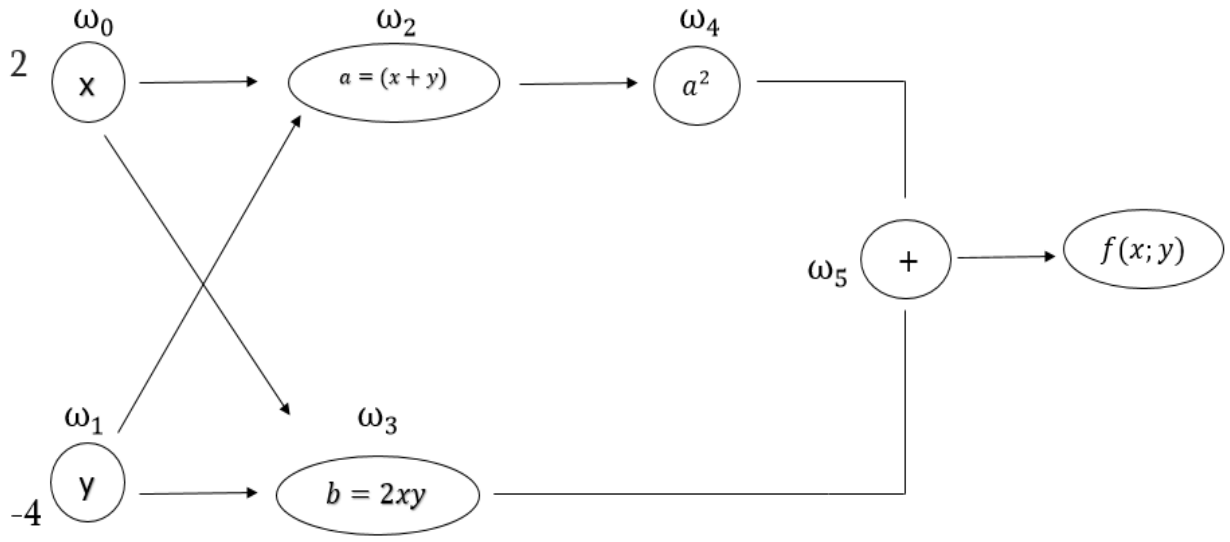


նկար 2.4

Ստացվեց, որ $\left.\frac{\partial f}{\partial x}\right|_{(2;-4)} = -12$, իսկ $\left.\frac{\partial f}{\partial y}\right|_{(2;-4)} = 0$: Այսինք հակադարձ մեթոդի դեպքում, շարժվում ենք ձախից դեպի աջ, և հաշվարկի արդյունքում ստանում ենք մասնակի ածանցյալների արժեքները բոլոր մուտքային տվյալների համար: Այս տարբերակը օգտագործվում է մեքենայական ուսուցման հետտարածման (backpropagation) խնդրում: Այժմ դիտարկենք ուղիղ մեթոդը:

<u>ՈՒղիղ մեթոդ:</u>

Առաջին քայլով յուրաքանչյուր գագաթ նշանակենք (նկ. 2.5) որևէ փոփոխականով, օրինակ. $\omega_0, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5$:



նկար 2.5

$\omega_0, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5$ –ը իրենցից ներկայացնում են x-ից կամ y-ից կախված ֆունկցիաներ.

$$\omega_0(x, y) = x$$

$$\omega_1(x, y) = y$$

$$\omega_2(x, y) = x + y$$

$$\dots$$

Այս մեթոդը յուրաքանչյուր ω_i-ին համապատասխանության մեջ է դնում $\dot{\omega}_i$ արժեքի հետ ՝ $(\omega_i; \dot{\omega}_i)$:

Այնուհետև հաշվենք ֆունկցիայի արժեքը տվյալ կետում, և ածանցյալները ըստ x-ի և y –ի:

ω_i -ի արժեքները՝

$\omega_0 = 2$

$\omega_1 = -4$

$\omega_2 = \omega_0 + \omega_1 = 2 - 4 = -2$

$\omega_3 = 2\omega_0\ \omega_1 = 2 * 2* (-4) = -16$

$\omega_4 = 2{\omega_2}^2 = 4$

$\omega_5 = \omega_3 + \omega_4 = 4 + (-16) = -12$

ածանցենք ըստ x -ի (ω_0 – ի)

$\dot{\omega}_0 = 1$

$\dot{\omega}_1 = 0$

$\dot{\omega}_2 = 1 * \dot{\omega}_0 = 1$

$\dot{\omega}_3 = 2\ \omega_1\ \dot{\omega}_0 = -8 *1 = -8$

$\dot{\omega}_4 = 2\ \omega_2\ \dot{\omega}_2 = -4 * 1 = -4$

$\dot{\omega}_5 = \dot{\omega}_3 + \dot{\omega}_4 = -8 -4 = -12$

ածանցենք ըստ y –ի (ω_1 - ի)

$\dot{\omega}_0 = 0$

$\dot{\omega}_1 = 1$

$\dot{\omega}_2 = 1 * \dot{\omega}_1 = 1$

$\dot{\omega}_3 = 2\ \omega_0\ \dot{\omega}_1 = 4$

$\dot{\omega}_4 = 2\ \omega_2\ \dot{\omega}_2 = -4 * 1 = -4$

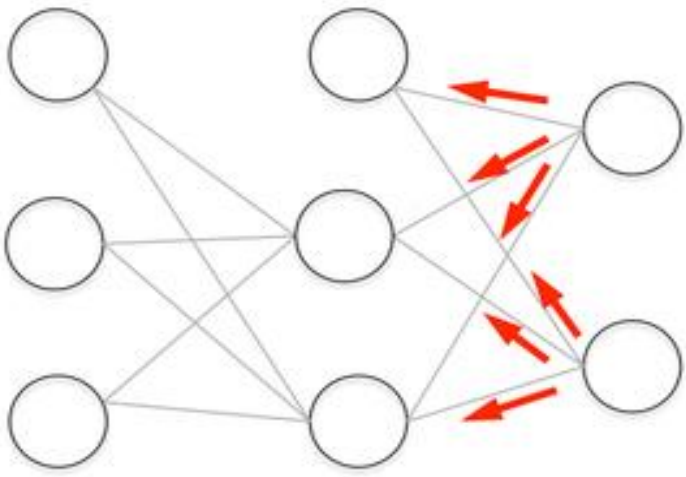
$\dot{\omega}_5 = \dot{\omega}_3 + \dot{\omega}_4 = -4 + 4 = 0$

Այսիքն այս մեթոդի դեպքում հաշվում և շարժվում ենք ածանցյալի մի ուղղությամբ : Ի տարբերություն հակադարձ մեթոդին, այս մեթոդի դեպում շարժվում ենք աջից ձախ:

Ավտոմատ դիֆերենցման այս երկու մեթոդները, լայնորեն կիրառվում են մեքենայական ուսուցման խնդիրներում, հատկապես հետտարածման (backpropagation algorithm) ալգորիթմում (նկ. 2.5), երբ անհրաժեշտ է լինում կորստի ֆունկցիան (loss function) նվազեցնելու կամ ձգտեցնելու 0-ի: Ենթադրենք, անհրաժեշտ է գտնել այնպիսի θ պարամետրերի բազմություն, որի դեպքում կորստի ֆունկցիան՝ $\mathrm{L}(\theta) \to \min$.

$$\theta_{t+1} = \theta_t - \alpha(\nabla \mathrm{L}(\theta))$$

որտեղ α-ն մոդելի ուսուցման արագությունն է : Ավտոմատ դիֆերենցումը ավտոմատ կերպով ապահովում է $\nabla L(\theta)$ արտահայտության ճշգրիտ հաշվարկը, անկախ մոդելի բարդությունից: Հիմնականում, խնդիրը կայանում է նրանում, որ կորստի ֆունկցիայի պարամետրերը թարմացնելու, փոփոխելու ժամանակ, անհրաժեշտ է լինում օգտվել գրադիենտային վայրէջքի մեթոդից, որի ժամանակ առաջանում է հարյուրավոր, երբեմն հազարավոր գրադիենտներ (մասնակի ածանցյալներ) հաշվելու անհրաժեշտություն: Ի տարբերություն թվային դիֆերենցմանը և սիմվոլիկ դիֆերենցմանը, ավտոմատ դիֆերենցումը շատ ավելի արդյունավետ է, քանի որ, շնորհիվ այս մեթոդի, այդ բոլոր գրադիենտները հաշվարկվում են մի քանի միլիվայրկյանների ընթացքում, և համեմատաբար շատ քիչ սխալներով, մինչ դեռ թվային դիֆերենցման ժամանակ առաջանում է սխալ, քանի որ ստանում ենք ածանցյալի արժեքի մոտավոր արժեքը, իսկ սիմվոլիկ դիֆերենցումը արդյունավետ է պարզ, տարրական ֆունկցիաների ժամանակ, այսինքն գործնական խնդիրներում առաջացնում է երբեմն դժվարություններ, և երկար ժամանակ է պահանջվում լուծելու համար:

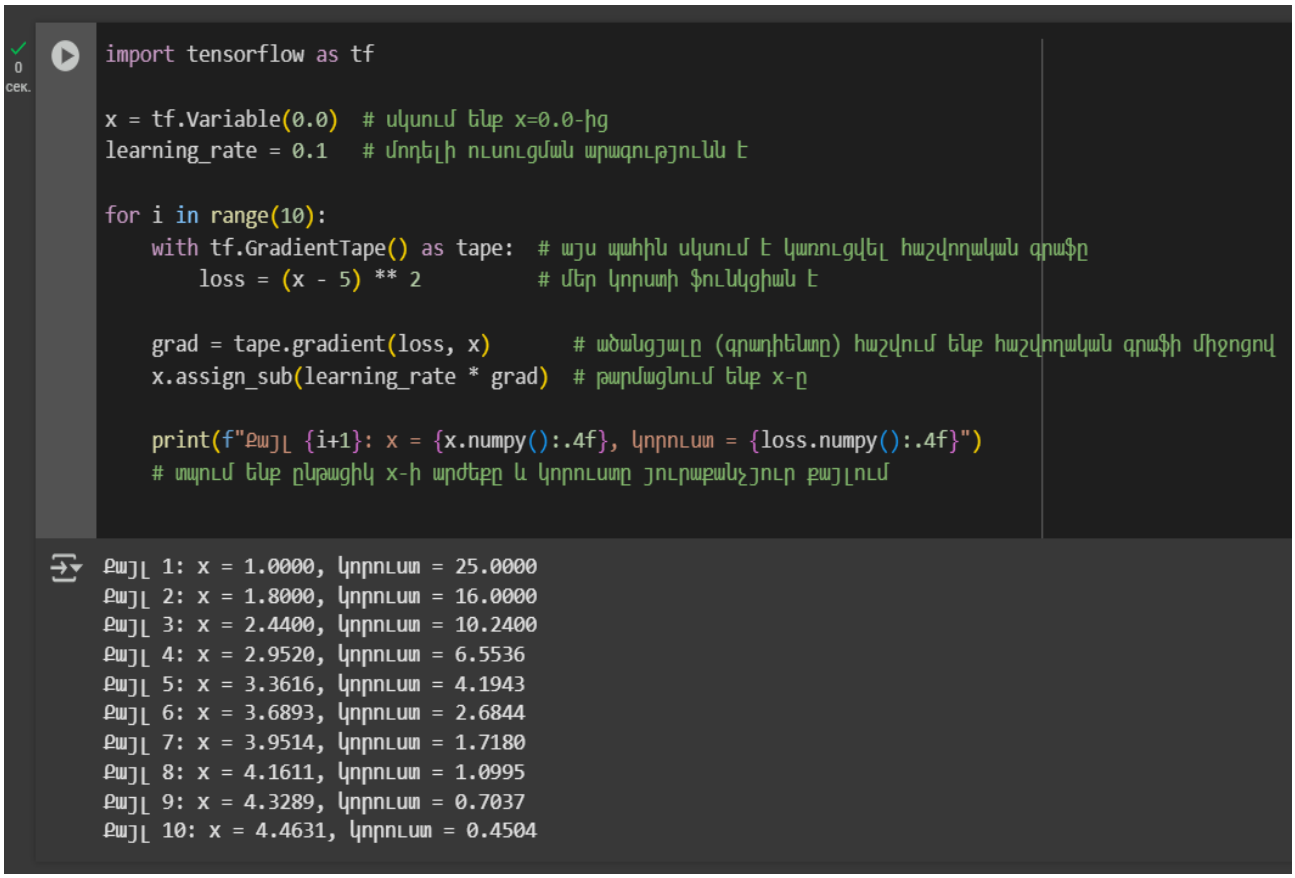


նկար 2.6

Մեթոդը իրագործվում է տարբեր ծրագրային փաթեթների, գրադարանների միջոցով, ինչպիսիք են TensorFlow-ը և PyTorch-ը : Դիտարկենք երկու օրինակներ, որոնք լուծվել են վերը նշված գրադարանների միջոցով, և ցույց տանք դրանց լուծման

քայլերի տարբերությունը և հաջորդականությունը: Ենթադրենք ունենք հետևյալ կորստի ֆունկցիան՝ $L(x) = (x-5)^2$

Օրինակ 1: TensorFlow

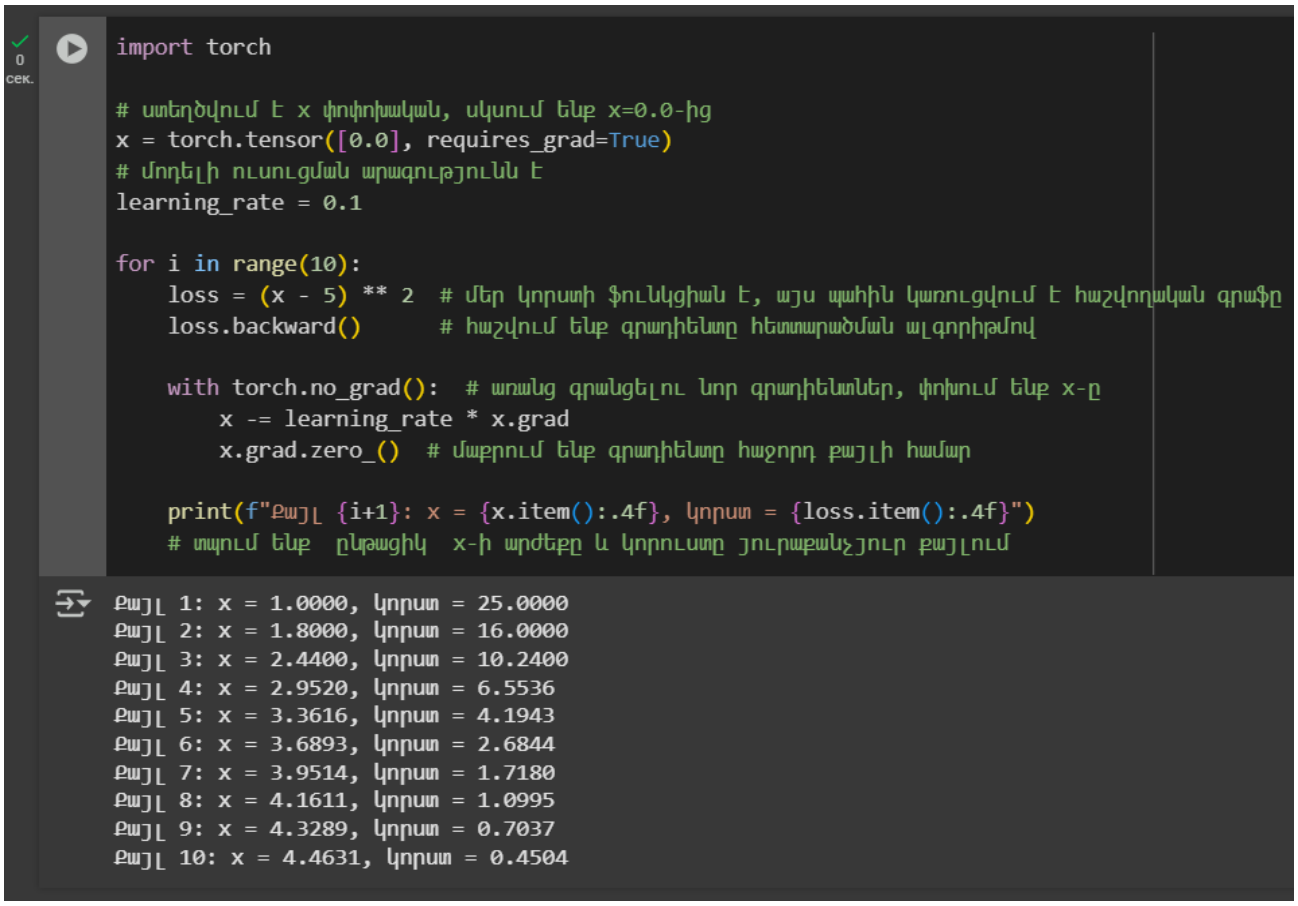


նկար 2.7

Ինչպես տեսնում ենք այստեղ նախորոք է կառուցվում հաշվողական գրաֆը, այսինքն ի սկզբանե սահմանում ենք բոլոր փոփոխականները, գործողությունները, ֆունկցիաները, որից հետո նոր հաշվում է գրադիենտները մինչև նշված քայլը, ու թարմացնում պարամետրերը: Այս մոտեցումը կոչվում է Static Graph կամ Define and Run: Այժմ դիտարկենք մյուս օրինակը: Ինչպես ցանկացած գրադարան կամ մեթոդ, այնպես էլ TensorFlow-ն ունի և՛ բացասական , և՛ դրական կողմեր: Այն ավելի հարմար է խոշոր նախագծերի համար, բայց Static Graph-ի պատճառով ավելի դժվար է դեբագ անել:

Օրինակ 2: PyTorch

Ի տարբերություն TensorFlow-ին, PyTorch-ում հաշվողական գրաֆը կառուցվում է ոչ թե մեկ անգամ և նախորոք, այլ ամեն անգան, երբ նոր թարմացված արժեքներով կատարվում են գործողություններ, այսինքն յուրաքանչյուր ցիկլում նոր գրաֆ է ստեղծվում դինամիկ կերպով: Այդ պատճառով է որ այս մոտեցմանը անվանում են Dynamic Graph կամ Define by Run:



```
import torch

# ստեղծվում է x փոփոխական, սկսում ենք x=0.0-ից
x = torch.tensor([0.0], requires_grad=True)
# մոդելի ուսուցման արագությունն է
learning_rate = 0.1

for i in range(10):
    loss = (x - 5) ** 2  # մեր կորստի ֆունկցիան է, այս պահին կառուցվում է հաշվողական գրաֆը
    loss.backward()      # հաշվում ենք գրադիենտը հետտարածման ալգորիթմով

    with torch.no_grad():  # առանց գրանցելու նոր գրադիենտներ, փոխում ենք x-ը
        x -= learning_rate * x.grad
        x.grad.zero_()  # մաքրում ենք գրադիենտը հաջորդ քայլի համար

    print(f"Քայլ {i+1}: x = {x.item():.4f}, կորստ = {loss.item():.4f}")
    # տպում ենք  ընթացիկ  x-ի արժեքը և կորուստը յուրաքանչյուր քայլում
```

```
Քայլ 1: x = 1.0000, կորստ = 25.0000
Քայլ 2: x = 1.8000, կորստ = 16.0000
Քայլ 3: x = 2.4400, կորստ = 10.2400
Քայլ 4: x = 2.9520, կորստ = 6.5536
Քայլ 5: x = 3.3616, կորստ = 4.1943
Քայլ 6: x = 3.6893, կորստ = 2.6844
Քայլ 7: x = 3.9514, կորստ = 1.7180
Քայլ 8: x = 4.1611, կորստ = 1.0995
Քայլ 9: x = 4.3289, կորստ = 0.7037
Քայլ 10: x = 4.4631, կորստ = 0.4504
```

նկար 2.8

Այսպիսով ծանոթացանք ավտոմատ դիֆերենցման մեթոդներին, որոցից յուրաքանչյուրը ունի իր ուրույն դերը ժամանակակից տեխնոլոգիաներում: Ինչպես տեսնում ենք ավտոմատ դիֆերենցման հակադարձ մեթոդի աշխատանքի սկզբունքը, ամբողջովին ընկած է PyTorch գրադարանում: Իսկ դուալ թվերը օգտագործվում է ավտոմատ դիֆերենցման ուղիղ մեթոդի ժամանակ: Կախված խնդրի դրվածքից, և

հասանելի (առկա) տվյալներից, յուրաքանչյուր մեթոդ կամ ալգորիթմ ունի իր տեխնիկական առավելությունները և թերությունները: Բայց միևնույն է, այս դիֆերենցման տարբերակը ամենալավագույնն է դիֆերենցման մյուս ալգորիթմների մեջ (թվային դիֆերենցում, սիմվոլիկ դեիֆերենցում):