



# PyTorch Autograd

גלעד מרקמן



קריית החינוך  
פארק המדע  
בית לערכים  
למצוינות ולחדשנות

# כללי



קריית החינוך  
פארק המדע  
בית לערכים  
למצוינות ולחדשנות

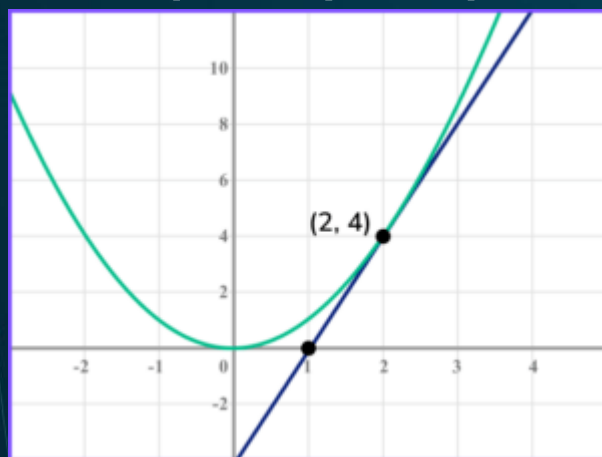
- מהותה של הנגזרת Gradient/Derivative והשימוש בה למציאת מינימום.
- חישוב נגזרות בסיסיות.
- נגזרת חלקית – נגזרת של פונקציה במספר משתנים.
- כלל השרשרת – נגזרת של פונקציה מורכבת.
- גרף חישוב ומציאת נגזרת באמצעות מחשב.
- ספריית torch.autograd לחישוב נגזרות.

[Colab Notebook](#)



# מהי נגזרת Gradient

- הנגזרת של גרף בנקודה מסויימת מתארת את השיפוע של הגרף (השיפוע של קו ישר המשיק לגרף בנקודה זו).

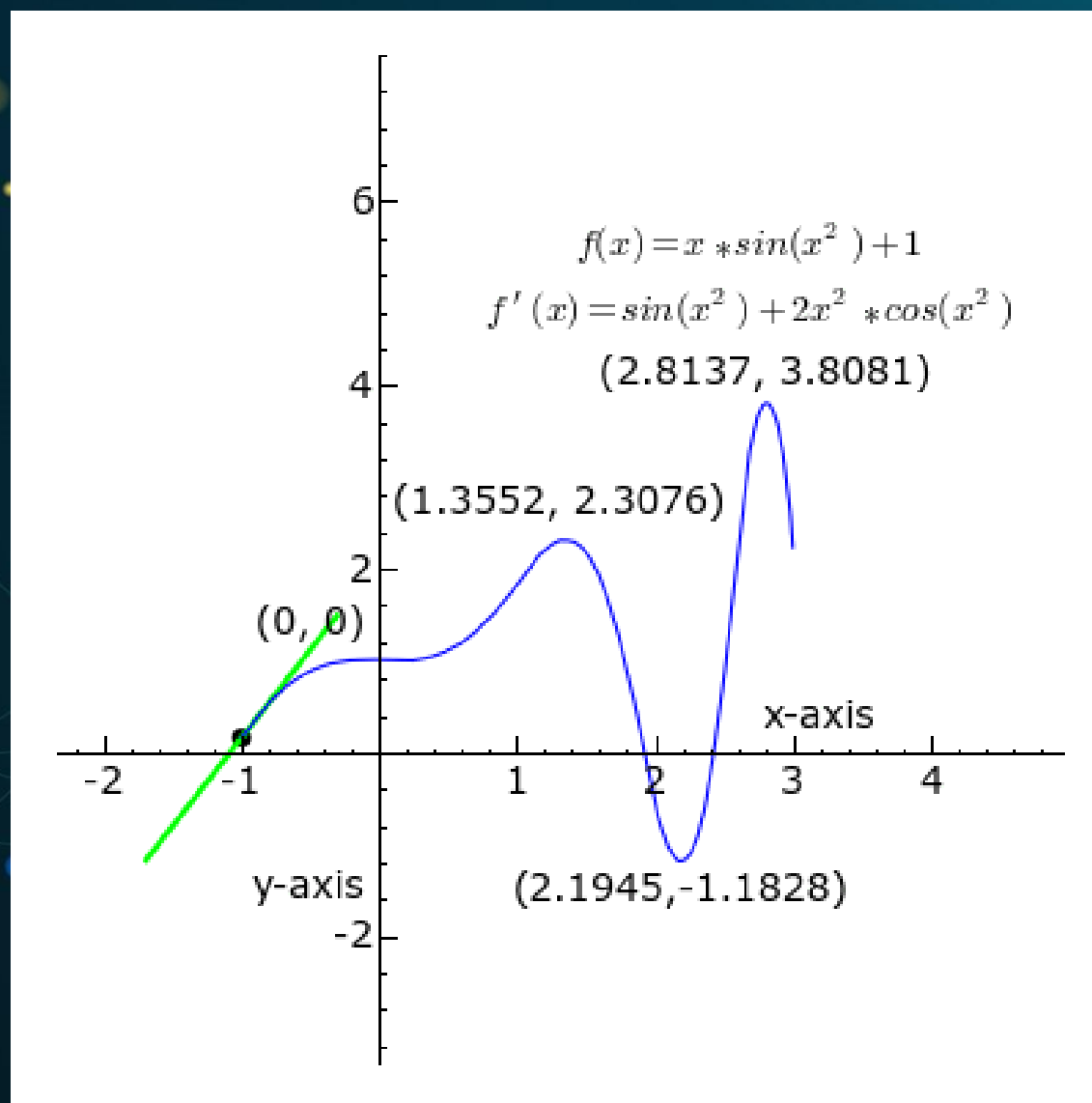


- אם הנגזרת חיובית – הפונקציה עולה בנקודה זו.
- אם הנגזרת שלילית – הפונקציה יורדת בנקודה זו.
- בנקודת מינימום/מקסימום – הנגזרת מתאפסת (משפט פרמה).

# השתנות הנגזרת בהתאם לשיפוע הגרף



קריית החינוך  
פארק המדע  
בית לערכים  
למציאות ולחדשנות





## חישוב נגזרת

• מספר כללים בסיסיים לחישוב הנגזרת.

• נסמן את הנגזרת של פונקציה  $f(x)$  ב  $f'(x)$ .

• נגזרת של קבוע היא 0:  $f(x) = 5; f'(x) = 0$

• נגזרת של  $x$  היא 1:  $f(x) = x; f'(x) = 1$

• נגזרת של כפל בקבוע:  $f(x) = k \cdot g(x); f'(x) = k \cdot g'(x)$

$$f(x) = 5x; f'(x) = 5$$

• נגזרת של סכום:  $f(x) = g(x) + h(x); f'(x) = g'(x) + h'(x)$

$$f(x) = 3x + 7; f'(x) = 3$$

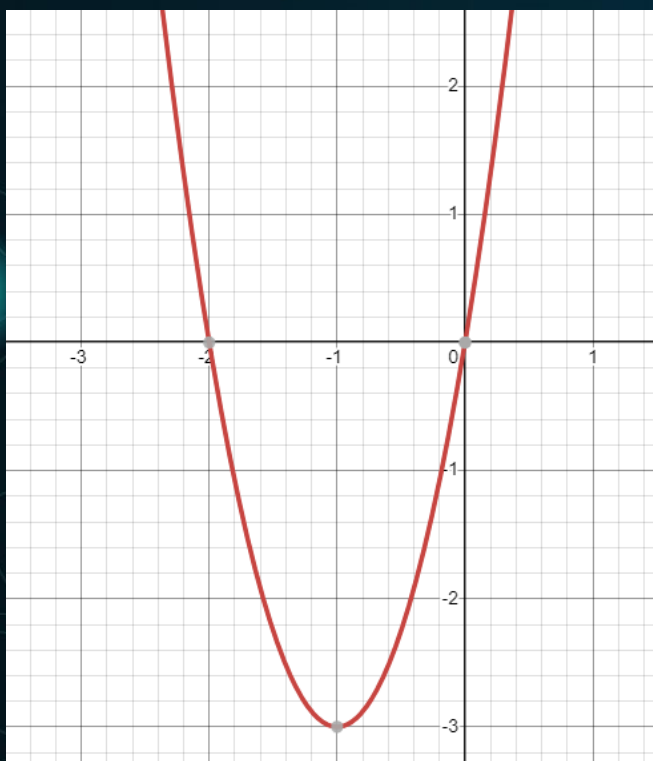
• נגזרת של פולינום:  $f(x) = x^n; f'(x) = nx^{n-1}$

$$f(x) = x^5; f'(x) = 5x^4$$



קריית החינוך  
פארק המדע  
בית לערכים  
למצוינות ולחדשנות

# תרגיל



• נתונה הפונקציה  $f(x) = 3x^2 + 6x$

• א. חשב את נוסחת הנגזרת

• ב. חשב את הנגזרת בנקודות:

$$x = 0, x = 2; x = -2$$

• ג. בכל אחת מנקודות אלה האם הפונקציה עולה או יורדת.

• ג. מצא נקודת המינימום של הפונקציה.



## נגזרת חלקית – פונקציה במספר משתנים

- כאשר נתונה פונקציה במספר משתנים ניתן לבצע גזירה חלקית של הפונקציה לפי כל אחד מהמשתנים.

- בנגזרת חלקית אנחנו מניחים כי המשתנים האחרים הם קבועים וגוזרים לפי המשתנה המבוקש בהתאם לאותם כללי גזירה.

- דוגמה:  $f(x,y) = 3x^2 + 2y^3 + 4xy$

- אם אנחנו גוזרים לפי  $x$  אז אנחנו מניחים ש  $y$  הוא קבוע. לכן, למשל, הנגזרת של  $2y^3$  היא 0 כיוון שכל המכפלה היא קבוע.

- $f'(x) = 6x + 0 + 4y = 6x + 4y$

- $f'(y) = 0 + 6y^2 + 4x = 6y^2 + 4x$



# סימונים לנגזרת

• נהוג לסמן נגזרות במספר סימונים מקובלים, בין היתר כדי להדגיש את המשתנה לפיו גוזרים

•  $f(x, y) = 3x^2 + 2y^3 + 4xy$

•  $f'(x) = \frac{df}{dx} = \frac{\Delta f}{\Delta x} = \frac{\partial f}{\partial x}$

•  $f'(y) = \frac{df}{dy} = \frac{\Delta f}{\Delta y} = \frac{\partial f}{\partial y}$

• כל הסימונים הללו הם זהים.





# כלל השרשרת

• כלל השרשרת נועד לעזור לנו בחישוב נגזרת של פונקציה מורכבת. למשל:

$$\bullet y = (3x^2 + 2)^3$$

• ניתן להציג זאת כפונקציה מורכבת משתי פונקציות, שכל אחת מהן אנחנו יודעים לגזור:

$$\bullet v = 3x^2 + 2; \frac{dv}{dx} = 6x$$

$$\bullet y = v^3; \frac{dy}{dv} = 3v^2$$

• כלל השרשרת קובע:

$$\bullet \frac{dy}{dx} = \frac{dy}{dv} \cdot \frac{dv}{dx} = 3v^2 \cdot 6x = 3(3x^2 + 2)^2 \cdot 6x = 18x(3x^2 + 2)^2$$



# כלל השרשרת המשך

• כאשר אנחנו גוזרים את הפונקציה מחוץ לסוגרים כפול הנגזרת הפנימית, אנחנו מפעילים באופן לא מודע את כלל השרשרת:

$$\bullet y = (3x^2 + 2)^3 = 3(3x^2 + 2)^2 \cdot 6x$$

$$\bullet \frac{dy}{dx} = \frac{dy}{dv} \cdot \frac{dv}{dx} = 3v^2 \cdot 6x = 3(3x^2 + 2)^2 \cdot 6x = 12x(3x^2 + 2)^2$$

• הכתיב של הנגזרת באמצעות חילוק עוזר לנו להדגים את כלל השרשרת כצמצום:

$$\bullet \frac{dy}{dx} = \frac{dy}{\cancel{dv}} \cdot \frac{\cancel{dv}}{dx}$$

• לא מדובר באמת בצמצום אלא זו דרך נוחה לייצג את הכלל.

# גרף חישוב



קריית החינוך  
פארק המדע  
בית לערכים  
למצוינות ולחדשנות

- חישוב נגזרת על ידי מחשב נעשית באמצעות גרף חישוב וכלל השרשרת.
- ניקח לדוגמה את הפונקציה  $l(x)$  הבאה, ונחלק לפונקציות משנה כך:

- $l = (4x - 1)^2$  ;

- $\hat{y} = 4x$  ;  $\frac{d\hat{y}}{dx} = 4$

- $l = (\hat{y} - 1)^2$ ;  $\frac{dl}{d\hat{y}} = 2(\hat{y} - 1)$

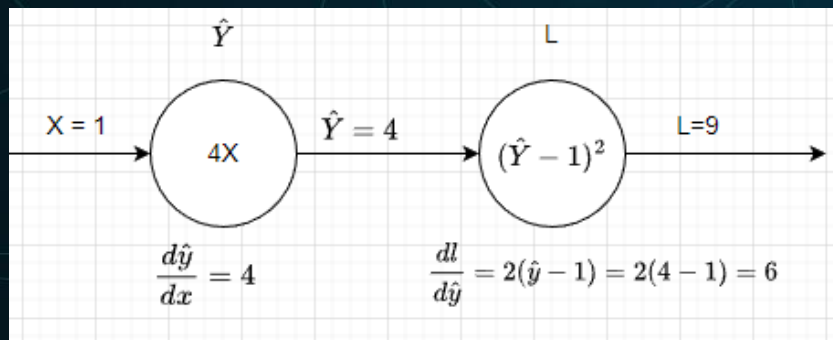
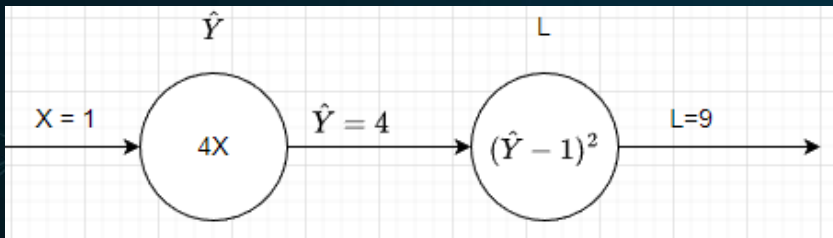
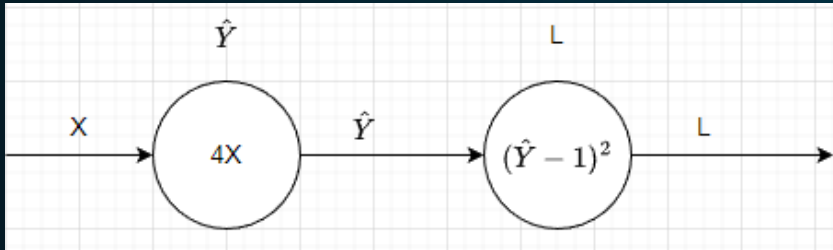
- $l'(x) = \frac{dl}{dx} = \frac{dl}{d\hat{y}} \cdot \frac{d\hat{y}}{dx} = 2(\hat{y} - 1) \cdot 4 = 2(4x - 1) \cdot 4 = 8(4x - 1)$

- $l'(1) = 8(4 \cdot 1 - 1) = 24$

# גרף חישוב

חישוב נגזרת במחשב נעשה בדרך הבאה:

• נבנה גרף חישוב כך שכל צומת היא פונקציה:



$$\frac{dl}{dx} = \frac{dl}{d\hat{y}} \cdot \frac{d\hat{y}}{dx} = 6 \cdot 4 = 24$$

• מבצע חישוב קדימה ל  $x = 1$  ונקבל תשובה 9:

• Forward()

• נבצע חישוב לאחור לצורך חישוב הנגזרת

• בנקודה  $x = 1$

backward()

# תרגילים



קריית החינוך  
פארק המדע  
בית לערכים  
למצוינות ולחדשנות

- לפי חישוב הנגזרת שעשינו בשקף הקודם מה תוכל לומר על הפונקציה  $l$  בנקודה  $x=1$ .
- בנה גרף חישוב לפונקציה  $l = (3x - 1)^2$  וחשב את הנגזרת בנקודה  $x=1$ . מה תוכל לומר על הפונקציה לפי הנגזרת.
- בנה גרף חישוב לפונקציה  $l = (1 \cdot x - 1)^2$  וחשב את הנגזרת בנקודה  $x=1$ . מה תוכל לומר כל הפונקציה לפי הנגזרת



# ספריית autograd.py

- ל-PyTorch יש ספרייה המאפשרת לחשב נגזרות בצורה אוטומטית.
- על מנת להפעיל את הספרייה עלינו להגדיר את המשתנה לפיו אנו גוזרים במאפיין `requires_grad=True`.
- כל חישוב שיעשה באמצעות Tensor זה תתווסף לו פעולה לחישוב הנגזרת.

```
x = torch.tensor([1.0], requires_grad=True)
v = 4 * x
z = v - 1
w = z ** 2
print (x)
print (v)
print (z)
print (w)
```

```
tensor([1.], requires_grad=True)
tensor([4.], grad_fn=<MulBackward0>)
tensor([3.], grad_fn=<SubBackward0>)
tensor([9.], grad_fn=<PowBackward0>)
```



# חישוב נגזרת באמצעות autograd

- הספרייה autograd מחשבת נגזרות באמצעות בניית גרף חישוב.
- החישוב של הנגזרת מחייב לבצע שני שלבים:
  - חילחול קדימה forward בו אנו מחשבים את תוצאות הפונקציה.
  - חילחול לאחור backward בו אנחנו מחשבים את הנגזרת בנקודה.

```
x = torch.tensor([1.0], requires_grad=True)
l = (4*x-1)**2 ←
print (l)
l.backward() ←
print (x.grad)
```

```
tensor([9.], grad_fn=<PowBackward0>)
tensor([24.])
```

Forward

Backward



# חישוב נגזרת חלקית במספר משתנים

- Pytorch יכולים גם לחשב נגזרת חלקית בפונקציה הכוללת מספר משתנים.
- נציג את אותה פונקציה שדנו בה קודם, אך הפעם במקום הקבועים נציג משתנים

```
x = torch.tensor([1.0], requires_grad=True)
w = torch.tensor([4.0], requires_grad=True)
y = torch.tensor([1.0])
loss = (w * x - y) ** 2
print(f"loss: {loss}")
loss.backward()
print (f"loss'(x=1) = {x.grad}")
print (f"loss'(w=4) = {w.grad}")
```

```
loss: tensor([9.], grad_fn=<PowBackward0>)
loss'(x=1) = tensor([24.])
loss'(w=4) = tensor([6.])
```

נגזרת של Loss לפי x בנקודה x=1

נגזרת של Loss לפי w בנקודה w=4





# נגזרת במספר נקודות בפונקציה המחזירה סקלר

- ניתן לחשב את הנגזרת במספר נקודות בפונקציה המקבלת מערך של ערכים (Tensor), אך מחזירה ערך בודד (סקלר).
- דוגמה לפונקציה המקבלת מספר ערכים ומחזירה ערך בודד היא פונקצית הממוצע `.mean`.
- במקרים אלו, ניתן לחשב את הנגזרת בכל הנקודות שהתקבלו:

```
X = torch.tensor([1,2,3,4], dtype=torch.float32, requires_grad=True)
w = torch.tensor([4.0, 4.0, 4.0, 4.0], requires_grad=True)
Y = torch.tensor([1.5,3,4.5,6])
```

```
L = ((w * X - Y) ** 2).mean()
print (L)
```

```
tensor(46.8750, grad_fn=<MeanBackward0>)
```

```
L.backward()
print (f"loss'(x) = {X.grad}")
print (f"loss'(w) = {w.grad}")
```

```
loss'(x) = tensor([ 5., 10., 15., 20.])
loss'(w) = tensor([ 1.2500,  5.0000, 11.2500, 20.0000])
```



# נגזרת במספר נקודות בפונקציה המחזירה ערכים מרובים

- כידוע ניתן לבצע חישובים מתמטיים על Tensor והתוצאה תהיה אף היא Tensor, דהיינו עם מספר ערכים.

```
X = torch.tensor([1,2,3,4], dtype=torch.float32, requires_grad=True)
w = torch.tensor([4.0, 4.0, 4.0, 4.0], requires_grad=True)
Y = torch.tensor([1.5,3,4.5,6])

L = (w * X - Y) ** 2
print (L)

tensor([ 6.2500, 25.0000, 56.2500, 100.0000], grad_fn=<PowBackward0>)
```

- חישוב הנגזרות מחייב להעביר לפונקציית backup וקטור שכל ערכיו 1 בגודל של הקלט. ללא הוקטור נקבל שגיאת הרצה.

```
v = torch.tensor([1.0,1.0,1.0,1.0])
L.backward(v)
print (f"loss'(x) = {X.grad}")
print (f"loss'(w) = {w.grad}")

loss'(x) = tensor([20., 40., 60., 80.])
loss'(w) = tensor([ 5., 20., 45., 80.])
```



# ביטול autograd

```
x = torch.tensor([1.0], requires_grad=True)
l = (4*x-1)**2
print (l)
l.backward()
print (x.grad)

y = x**2
print (y)
y.backward()
print (x.grad)

tensor([9.], grad_fn=<PowBackward0>)
tensor([24.])
tensor([1.], grad_fn=<PowBackward0>)
tensor([26.])
```

- בכל פעולה מתמטית שאנו מבצעים חישוב הנגזרת נצברת ב Tensor, דבר זה עלול לגרום לחישובים לא נכונים.

- בנוסף לכך, Tensor הכולל גרדיאנים מוגבל בפעולותיו, לדוגמה לא ניתן להפוך אותו למערך numpy.

```
x_np = x.numpy()
```

```
-----
RuntimeError                                Traceback (most recent call last)
<ipython-input-18-3eadde1473e0> in <module>
----> 1 x_np = x.numpy()
```

```
RuntimeError: Can't call numpy() on Tensor that requires grad. Use tensor.detach().numpy() instead.
```



# ביטול autograd

ניתן לבטל / לנתק את autograd מ Tensor באחת מהדרכים הבאות:

```
x = torch.tensor([1.0], requires_grad=True)
print(x)
x.requires_grad_(False)
print(x)
```

```
tensor([1.], requires_grad=True)
tensor([1.])
```

- שינוי מאפיין requires\_grad לערך False

- העתקה ל- Tensor חדש ללא autograd באמצעות detach().

```
x = torch.tensor([1.0], requires_grad=True)
print(x)
y = x.detach()
print(y)
```

```
tensor([1.], requires_grad=True)
tensor([1.])
```



# השהיית autograd

- שימוש בפקודה `with torch.no_grad()` המששה באופן זמני את חישוב הגרדיאנים.

```
x = torch.tensor([1.0], requires_grad=True)
with torch.no_grad():
    x_np = x.numpy()
print (x_np)
```

```
[1.]
```



# איפוס הגרדיאנים

- כאמור, בכל חלחול לאחור (`backward()`) הגרדיאנים נצברים בטנסור. דבר זה עלול לגרום לטעויות חישוב ולכן נרצה לאפס את הגרדיאנים.

```
W = torch.tensor([2.0, 3, 5, 8], requires_grad=True)
```

```
for epoch in range(3):  
    model = (W * 3).mean()  
    model.backward()  
    print (W.grad)  
    #W.grad.zero_()
```

```
tensor([0.7500, 0.7500, 0.7500, 0.7500])  
tensor([1.5000, 1.5000, 1.5000, 1.5000])  
tensor([2.2500, 2.2500, 2.2500, 2.2500])
```

```
W = torch.tensor([2.0, 3, 5, 8], requires_grad=True)
```

```
for epoch in range(3):  
    model = (W * 3).mean()  
    model.backward()  
    print (W.grad)  
    W.grad.zero_()
```

```
tensor([0.7500, 0.7500, 0.7500, 0.7500])  
tensor([0.7500, 0.7500, 0.7500, 0.7500])  
tensor([0.7500, 0.7500, 0.7500, 0.7500])
```



# תרגיל

• חשב בעזרת pytorch את הנגזרות של הפונקציות הבאות בנקודות הבאות:

- $y = 3x^2 + \frac{1}{2}x + 5$  ;  $x = -3, 1, 4, 9$
- $loss = ((W \cdot X - Y) ** 2).mean( )$  ;
  - $W = 0, -0.5, 1, 2, 3$ ;  $X = 1, 2, 3, 4$ ;  $Y = 2, 4, 6, 8$
  - מצא את הנגזרות של הפונקציה לפי W
- $f = 2x^3 - 3x^2$  ;  $-2, 0, 0.5, 1, 2$ 
  - מה אתה למד מהנגזרות