[DAN.IT] EDUCATION

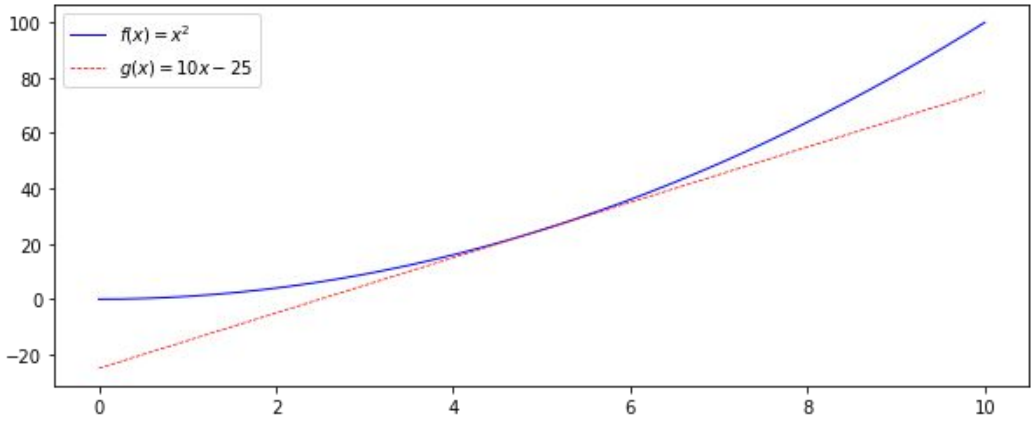
Machine learning

Session 4 - Calculus and matrices

Univariate derivatives

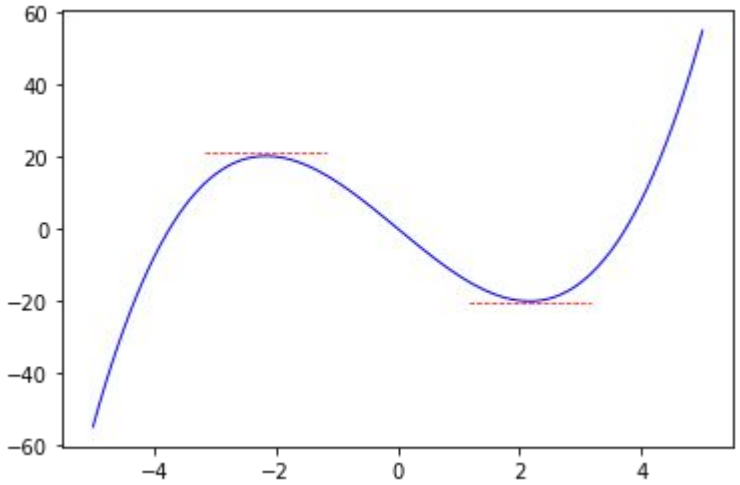
f(x) describes a function of x, i.e. gives us information about the function for some x.

The **derivative** of f(x) gives us information about how the function changes (locally) around x. Specifically, the slope of the tangent to f(x) at x.



Extremum points

By finding values of x where the derivative of f(x) is 0, we can find local extremum points (minima and maxima) of the function.

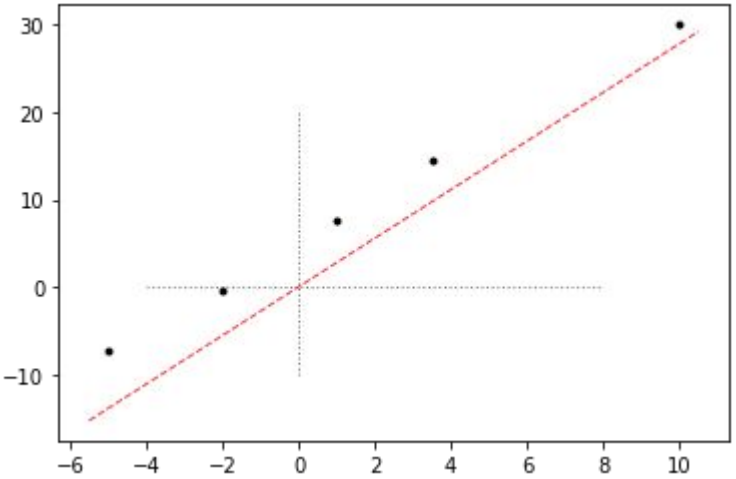


Derivation rules

Rule	$f(x)$	Scalar derivative notation with respect to x	Example
Constant	c	0	$\frac{d}{dx}99 = 0$
Multiplication by constant	cf	$c\frac{df}{dx}$	$\frac{d}{dx}3x = 3$
Power Rule	x^n	nx^{n-1}	$\frac{d}{dx}x^3 = 3x^2$
Sum Rule	$f + g$	$\frac{df}{dx} + \frac{dg}{dx}$	$\frac{d}{dx}(x^2 + 3x) = 2x + 3$
Difference Rule	$f - g$	$\frac{df}{dx} - \frac{dg}{dx}$	$\frac{d}{dx}(x^2 - 3x) = 2x - 3$
Product Rule	fg	$f\frac{dg}{dx} + \frac{df}{dx}g$	$\frac{d}{dx}x^2x = x^2 + x2x = 3x^2$
Chain Rule	$f(g(x))$	$\frac{df(u)}{du}\frac{du}{dx}$, let $u = g(x)$	$\frac{d}{dx}ln(x^2) = \frac{1}{x^2}2x = \frac{2}{x}$

Toy “linear regression”

Find the line that passes through the origin and best fits the following points:
[(1, 7.5), (-2, -0.4), (3.5, 14.5), (10, 30), (-5, -7.3)]



$$\arg\min_{m} \sum_{i=1}^{n} \|m \cdot x_i - y_i\|_2$$

Toy “linear regression”

$$L(m) = \sum_{i=1}^{5} \|m \cdot x_i - y_i\|_2 = \sum_{i=1}^{5} (m \cdot x_i - y_i)^2$$

$$= (m - 7.5)^2 + (-2m + 0.4)^2 + (3.5m - 14.5)^2 + (10m - 30)^2 + (-5m + 7.3)^2$$

Toy “linear regression”

$$\frac{d}{dm}L(m) = 2(m - 7.5)(1) + 2(-2m + 0.4)(-2) +$$

$$+ 2(3.5 - 14.5)(3.5) + 2(10m - 30)(10) + 2(-5m + 7.3)(-5)$$

$$= \ldots = 284.5m - 791.1$$

$$\frac{d}{dm}L(m) = 0 \Longrightarrow 284.5m = 791.1 \Longrightarrow$$

$$m = 2.78066\ldots$$

Partial derivatives

For **multivariate functions**, the partial derivative with respect to some variable gives us information about how the function changes as that variable changes (again, locally).
The partial derivative is like the univariate derivative treating all other variables as constants.

$$f(x, y, z) = x^3y^2 + z$$

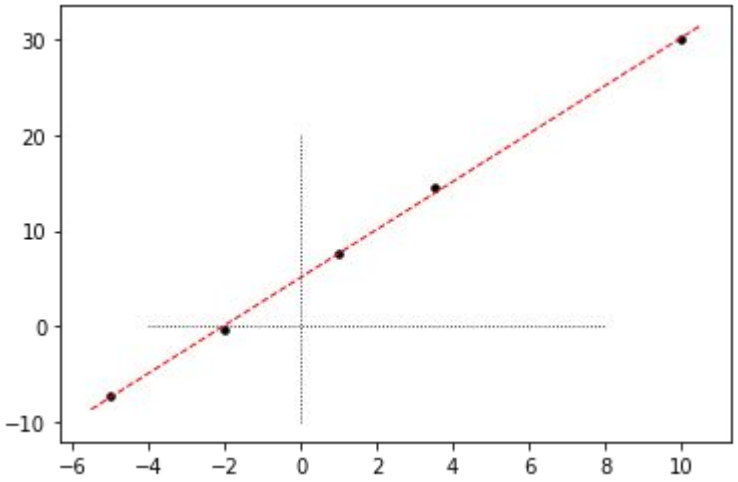
$$\frac{\partial f}{\partial x} = 3x^2y^2$$

$$\frac{\partial f}{\partial y} = 2x^3y$$

$$\frac{\partial f}{\partial z} = 1$$

Simple linear regression

Find the line that best fits the following points: [(1, 7.5), (-2, -0.4), (3.5, 14.5), (10, 30), (-5, -7.3)]



$$\arg\min_{m,n} \sum_{i=1}^{k} \|m \cdot x_i + n - y_i\|_2$$

Simple linear regression

$$L(m, n) = \sum_{i=1}^{k} \|m \cdot x_i + n - y_i\|_2 = \sum_{i=1}^{5} (m \cdot x_i + n - y_i)^2$$

$$\frac{\partial L}{\partial m} = \sum_{i=1}^{5} 2(m \cdot x_i + n - y_i)(x_i) = \ldots = 284.5m + 15n - 791.1$$

$$\frac{\partial L}{\partial n} = \sum_{i=1}^{5} 2(m \cdot x_i + n - y_i)(1) = 15m + 10n - 88.6$$

Simple linear regression

$$\frac{\partial L}{\partial m} = 0, \frac{\partial L}{\partial n} = 0 \Longrightarrow$$

$$284.5m + 15n - 791.1 = 0$$

$$15m + 10n - 88.6 = 0$$

$$\ldots \Longrightarrow m \approx 2.5122,\ n \approx 5.091$$

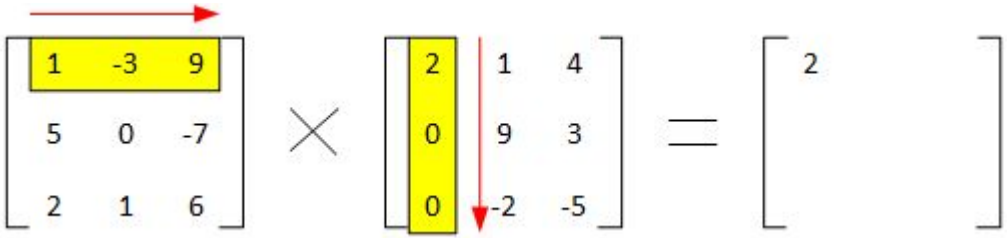
Quick linear algebra refresher

Linear algebra mostly deals with vectors, matrices, and operations between them.

$$M = \begin{bmatrix} 2 & 1 & -1 \\ -3 & -1 & 2 \\ -2 & 1 & 2 \end{bmatrix} \qquad v = \begin{bmatrix} 8 \\ -11 \\ -3 \end{bmatrix}$$

Quick linear algebra refresher

Matrix multiplication is a common, useful operation between two matrices (of appropriate sizes).



Matrix multiplication is not commutative, i.e. A . B != B . A.

Quick linear algebra refresher

The **transpose** of a matrix A is another matrix, whose column vectors are A's row vectors (or vice versa).

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix} = \begin{bmatrix} 1 & 5 & 9 \\ 2 & 6 & 10 \\ 3 & 7 & 11 \\ 4 & 8 & 12 \end{bmatrix}^{\mathrm{T}}$$

Quick linear algebra refresher

The **Identity matrix** of order n is an nxn matrix whose main diagonal is 1 and all other elements are 0.

$$I_1 = [1],\ I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},\ \cdots,\ I_n = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

Quick linear algebra refresher

The **inverse** of a square matrix is another matrix that, when multiplied together, yield the identity matrix.

$$A \cdot A^{-1} = A^{-1} \cdot A = I$$

Not all square matrices are invertible.

Gradient

The **gradient** of a multivariate function is a vector field composed of the partial derivatives of the function.

$$f(x, y, z) = x^3 y^2 + z$$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{bmatrix} = \begin{bmatrix} 3x^2 y^2 \\ 2x^3 y \\ 1 \end{bmatrix}$$

Jacobian

Suppose we have several multivariate functions. The **Jacobian** is a matrix for organizing all the partial derivatives of all the functions; it is constructed by stacking all the (transposes of the) gradients*.

$$J = \begin{bmatrix} \nabla f(x,y) \\ \nabla g(x,y) \end{bmatrix} = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} & \frac{\partial f(x,y)}{\partial y} \\ \frac{\partial g(x,y)}{\partial x} & \frac{\partial g(x,y)}{\partial y} \end{bmatrix}$$

* This notation is called the numerator layout; the denominator layout is simply the transpose.

Jacobian

For simpler notation we will refer to multivariate parameters as a single vector, e.g. **x**, and to the collection of functions as a vector (of functions), e.g. **y**.

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \nabla f_1(\mathbf{x}) \\ \nabla f_2(\mathbf{x}) \\ \dots \\ \nabla f_m(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial \mathbf{x}} f_1(\mathbf{x}) \\ \frac{\partial}{\partial \mathbf{x}} f_2(\mathbf{x}) \\ \dots \\ \frac{\partial}{\partial \mathbf{x}} f_m(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x_1} f_1(\mathbf{x}) & \frac{\partial}{\partial x_2} f_1(\mathbf{x}) & \dots & \frac{\partial}{\partial x_n} f_1(\mathbf{x}) \\ \frac{\partial}{\partial x_1} f_2(\mathbf{x}) & \frac{\partial}{\partial x_2} f_2(\mathbf{x}) & \dots & \frac{\partial}{\partial x_n} f_2(\mathbf{x}) \\ & & \dots & \\ \frac{\partial}{\partial x_1} f_m(\mathbf{x}) & \frac{\partial}{\partial x_2} f_m(\mathbf{x}) & \dots & \frac{\partial}{\partial x_n} f_m(\mathbf{x}) \end{bmatrix}$$

Matrix calculus

The matrix calculus we'll introduce is quite light, and deals with rules that simplify the computation of Jacobians (instead of having to calculate each partial derivative element of the Jacobian separately).

Vector element-wise operations

Using ○ to denote elementwise operations, suppose we have:

$$\mathbf{y} = \mathbf{f}(\mathbf{w}) \bigcirc \mathbf{g}(\mathbf{x})$$

i.e.:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x}) \\ f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x}) \end{bmatrix}$$

Vector element-wise operations

By definition of the Jacobian (w.r.t. **w** in this case):

$$J_{\mathbf{w}} = \frac{\partial \mathbf{y}}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial}{\partial w_1}(f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x})) & \frac{\partial}{\partial w_2}(f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x})) & \dots & \frac{\partial}{\partial w_n}(f_1(\mathbf{w}) \bigcirc g_1(\mathbf{x})) \\ \frac{\partial}{\partial w_1}(f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x})) & \frac{\partial}{\partial w_2}(f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x})) & \dots & \frac{\partial}{\partial w_n}(f_2(\mathbf{w}) \bigcirc g_2(\mathbf{x})) \\ & \dots & & \\ \frac{\partial}{\partial w_1}(f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x})) & \frac{\partial}{\partial w_2}(f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x})) & \dots & \frac{\partial}{\partial w_n}(f_n(\mathbf{w}) \bigcirc g_n(\mathbf{x})) \end{bmatrix}$$

Since the operation is elementwise, $y_i = f_i(\mathbf{w}) \bigcirc g_i(\mathbf{x})$ will depend only on w_i, x_i.

Therefore:

$$\frac{\partial \mathbf{y}}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial}{\partial w_1}(f_1(w_1) \bigcirc g_1(x_1)) & & & \\ & \frac{\partial}{\partial w_2}(f_2(w_2) \bigcirc g_2(x_2)) & & 0 \\ & & \dots & \\ 0 & & & \frac{\partial}{\partial w_n}(f_n(w_n) \bigcirc g_n(x_n)) \end{bmatrix}$$

$$= diag\left(\frac{\partial}{\partial w_1}(f_1(w_1) \bigcirc g_1(x_1)),\ \frac{\partial}{\partial w_2}(f_2(w_2) \bigcirc g_2(x_2)),\ \dots,\ \frac{\partial}{\partial w_n}(f_n(w_n) \bigcirc g_n(x_n))\right)$$

Vector element-wise operations

Op	Partial with respect to w
$+$	$\frac{\partial(\mathbf{w}+\mathbf{x})}{\partial \mathbf{w}} = diag(\ldots \frac{\partial(w_i+x_i)}{\partial w_i} \ldots) = diag(\vec{1}) = I$
$-$	$\frac{\partial(\mathbf{w}-\mathbf{x})}{\partial \mathbf{w}} = diag(\ldots \frac{\partial(w_i-x_i)}{\partial w_i} \ldots) = diag(\vec{1}) = I$
$\otimes$	$\frac{\partial(\mathbf{w}\otimes\mathbf{x})}{\partial \mathbf{w}} = diag(\ldots \frac{\partial(w_i\times x_i)}{\partial w_i} \ldots) = diag(\mathbf{x})$
$\oslash$	$\frac{\partial(\mathbf{w}\oslash\mathbf{x})}{\partial \mathbf{w}} = diag(\ldots \frac{\partial(w_i/x_i)}{\partial w_i} \ldots) = diag(\ldots \frac{1}{x_i} \ldots)$

Op	Partial with respect to x
$+$	$\frac{\partial(\mathbf{w}+\mathbf{x})}{\partial \mathbf{x}} = I$
$-$	$\frac{\partial(\mathbf{w}-\mathbf{x})}{\partial \mathbf{x}} = diag(\ldots \frac{\partial(w_i-x_i)}{\partial x_i} \ldots) = diag(-\vec{1}) = -I$
$\otimes$	$\frac{\partial(\mathbf{w}\otimes\mathbf{x})}{\partial \mathbf{x}} = diag(\mathbf{w})$
$\oslash$	$\frac{\partial(\mathbf{w}\oslash\mathbf{x})}{\partial \mathbf{x}} = diag(\ldots \frac{-w_i}{x_i^2} \ldots)$

Vector sum reduction

Suppose $y = sum(\mathbf{f}(\mathbf{x})) = \sum_{i=1}^{n} f_i(\mathbf{x})$:

$$
\begin{aligned}
\nabla y &= \left[\frac{\partial y}{\partial x_1}, \frac{\partial y}{\partial x_2}, \ldots, \frac{\partial y}{\partial x_n}\right] \\
&= \left[\frac{\partial}{\partial x_1}\sum_i f_i(\mathbf{x}),\ \frac{\partial}{\partial x_2}\sum_i f_i(\mathbf{x}),\ \ldots,\ \frac{\partial}{\partial x_n}\sum_i f_i(\mathbf{x})\right] \\
&= \left[\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_1},\ \sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_2},\ \ldots,\ \sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_n}\right]
\end{aligned}
$$

Vector sum reduction

For the simple case of $y = sum(\mathbf{x})$ (that is, $f_i(\mathbf{x}) = x_i$):

$$\nabla y = \left[\sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_1}, \ \sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_2}, \ \ldots, \ \sum_i \frac{\partial f_i(\mathbf{x})}{\partial x_n}\right] = \left[\sum_i \frac{\partial x_i}{\partial x_1}, \ \sum_i \frac{\partial x_i}{\partial x_2}, \ \ldots, \ \sum_i \frac{\partial x_i}{\partial x_n}\right]$$

$$= \left[\frac{\partial x_1}{\partial x_1}, \ \frac{\partial x_2}{\partial x_2}, \ \ldots, \ \frac{\partial x_n}{\partial x_n}\right] = [1, 1, \ldots, 1] = \vec{1}^T$$

Vector chain rule

A generalization of the chain rule to matrices:

$$\frac{\partial}{\partial x}\mathbf{f}(\mathbf{g}(x)) = \frac{\partial \mathbf{f}}{\partial \mathbf{g}}\frac{\partial \mathbf{g}}{\partial x}$$

$$= \begin{bmatrix} \frac{\partial f_1}{\partial g_1} & \frac{\partial f_1}{\partial g_2} & \cdots & \frac{\partial f_1}{\partial g_k} \\ \frac{\partial f_2}{\partial g_1} & \frac{\partial f_2}{\partial g_2} & \cdots & \frac{\partial f_2}{\partial g_k} \\ & & \cdots & \\ \frac{\partial f_m}{\partial g_1} & \frac{\partial f_m}{\partial g_2} & \cdots & \frac{\partial f_m}{\partial g_k} \end{bmatrix} \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ & & \cdots & \\ \frac{\partial g_k}{\partial x_1} & \frac{\partial g_k}{\partial x_2} & \cdots & \frac{\partial g_k}{\partial x_n} \end{bmatrix}$$

This generalization is not at all trivial; see reference for details and proof.

Linear regression (OLS)

Let's formulate linear regression (ordinary least squares) as vector multiplication, padding **x** with a 1 (for the intercept):

$$y = a_0 + a_1 \cdot x_1 + a_2 \cdot x_2 + \ldots + a_n \cdot x_n = \mathbf{a} \cdot \mathbf{x}$$

Over a set of pairs $\mathbf{x_i}$, y_i we want to choose **a** so as to minimize the sum of squared errors:

$$E(\mathbf{a}, \mathbf{x_i}, y_i) = \sum_i (y_i - \mathbf{a} \cdot \mathbf{x_i})^2$$

Linear regression (OLS)

Let’s find the partial derivative of E w.r.t **a**; we’ll use some substitutions.

$$u(\mathbf{a}, \mathbf{x}) = \mathbf{a} \cdot \mathbf{x}$$

$$v(u, y) = y - u$$

$$E(v) = \sum_i v_i^2$$

Linear regression (OLS)

We'll note that dot product is a vector sum reduction of element-wise multiplication. Therefore due to:

1. Partial derivative of element-wise multiplication;
2. Gradient of vector sum reduction; and
3. Vector chain rule

We obtain:

$$\frac{\partial u}{\partial \mathbf{a}} = \vec{1}^{T} diag(\mathbf{x}) = \mathbf{x}^{T}$$

Linear regression (OLS)

$$\frac{\partial v}{\partial \mathbf{a}} = \frac{\partial}{\partial \mathbf{a}}(y - u) = \vec{0}^T - \frac{\partial u}{\partial \mathbf{a}} = -\mathbf{x}^T$$

$$\frac{\partial E}{\partial \mathbf{a}} = \frac{\partial}{\partial \mathbf{a}} \sum_i v_i^2 = \sum_i \frac{\partial v_i^2}{\partial \mathbf{a}} = \sum_i \frac{\partial v_i^2}{\partial v_i} \frac{\partial v_i}{\partial (\mathbf{a})} = \sum_i 2 v_i \cdot -\mathbf{x_i}^T =$$

$$= -2 \sum_i (y_i - \mathbf{a} \cdot \mathbf{x_i}) \cdot \mathbf{x_i}^T = 2 \sum_i (\mathbf{a} \cdot \mathbf{x_i} - y_i) \cdot \mathbf{x_i}^T$$

Linear regression (OLS)

$$\frac{\partial E}{\partial \mathbf{a}} = 2\sum_i (\mathbf{a}\cdot\mathbf{x_i} - y_i)\cdot \mathbf{x_i}^T = 2\begin{pmatrix} (\mathbf{a}\cdot\mathbf{x_1} - y_1)\cdot \mathbf{x_1}^T \\ + \\ (\mathbf{a}\cdot\mathbf{x_2} - y_2)\cdot \mathbf{x_2}^T \\ + \\ \dots \\ + \\ (\mathbf{a}\cdot\mathbf{x_n} - y_n)\cdot \mathbf{x_n}^T \end{pmatrix}$$

$$= 2\begin{bmatrix} (\mathbf{a}\cdot\mathbf{x_1} - y_1)\cdot x_{11} + (\mathbf{a}\cdot\mathbf{x_2} - y_2)\cdot x_{21} + \ldots + (\mathbf{a}\cdot\mathbf{x_n} - y_n)\cdot x_{n1} \\ (\mathbf{a}\cdot\mathbf{x_1} - y_1)\cdot x_{12} + (\mathbf{a}\cdot\mathbf{x_2} - y_2)\cdot x_{22} + \ldots + (\mathbf{a}\cdot\mathbf{x_n} - y_n)\cdot x_{n2} \\ \dots \\ (\mathbf{a}\cdot\mathbf{x_1} - y_1)\cdot x_{1m} + (\mathbf{a}\cdot\mathbf{x_2} - y_2)\cdot x_{2m} + \ldots + (\mathbf{a}\cdot\mathbf{x_n} - y_n)\cdot x_{nm} \end{bmatrix}^T$$

Linear regression (OLS)

$$= 2 \begin{bmatrix} \mathbf{a} \cdot \mathbf{x_1} - y_1 \\ \mathbf{a} \cdot \mathbf{x_2} - y_2 \\ \cdots \\ \mathbf{a} \cdot \mathbf{x_n} - y_n \end{bmatrix}^T \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \cdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}$$

$$= (\mathbf{a}^T \underbrace{\begin{bmatrix} x_{11} & x_{21} & \cdots & x_{n1} \\ x_{12} & x_{22} & \cdots & x_{n2} \\ \cdots & & & \\ x_{1m} & x_{2m} & \cdots & x_{nm} \end{bmatrix}}_{X^T} - \underbrace{\begin{bmatrix} y_1 & y_2 & \cdots & y_n \end{bmatrix}}_{\mathbf{y}^T}) \underbrace{\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \cdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}}_{X}$$

Linear regression (OLS)

If we reorganize all $\mathbf{x}_i$ in a matrix and y_i as a column vector:

$$X = \begin{bmatrix} -- \mathbf{x_1}^T -- \\ -- \mathbf{x_2}^T -- \\ \cdots \\ -- \mathbf{x_n}^T -- \end{bmatrix}; \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdots \\ y_n \end{bmatrix}$$

Then we get:

$$\frac{\partial E(\mathbf{a}, X, \mathbf{y})}{\partial \mathbf{a}} = 2(\mathbf{a}^T X^T - \mathbf{y}^T) \cdot X$$

Linear regression (OLS)

From here we can find the analytical solution by equating the gradient to 0:

$$(\mathbf{a}^T X^T - \mathbf{y}^T) \cdot X = \mathbf{0}$$

$$\mathbf{a}^T X^T X - \mathbf{y}^T X = \mathbf{0}$$

$$\mathbf{a}^T X^T X = \mathbf{y}^T X$$

$$(\mathbf{a}^T X^T X)^T = (\mathbf{y}^T X)^T$$

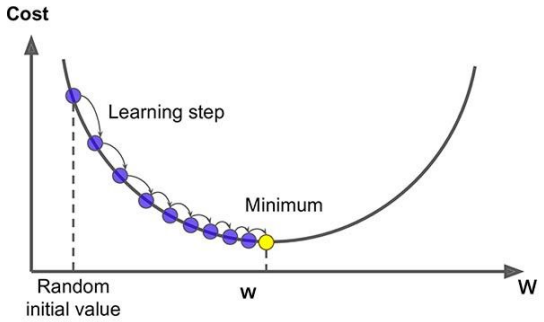
$$(X^T X)^T \mathbf{a} = X^T \mathbf{y}$$

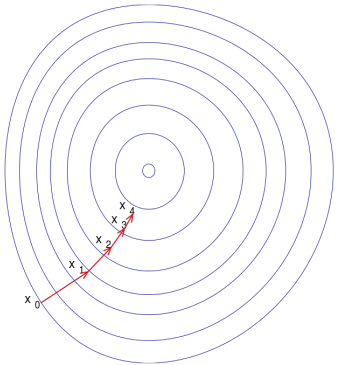
$$X^T X \mathbf{a} = X^T \mathbf{y}$$

$$\mathbf{a} = (X^T X)^{-1} X^T \mathbf{y}$$

Basic gradient descent

The analytical solution to OLS requires inverting a matrix, which could be infeasible for (very) large matrices. An optimization technique for differentiable problems for which an analytical solutions are infeasible is called **gradient descent**, and involves taking small steps in the **direction of greatest descent** (or ascent).





Basic gradient descent

A simple algorithm sketch:

```
OLS_gradient_descent(X, y, step_size, termination_criteria):
    initialize random a
    while termination_criteria isn't met:
        current_gradient ← Calculate current gradient
        a ← a - step_size * current_gradient
    return a
```

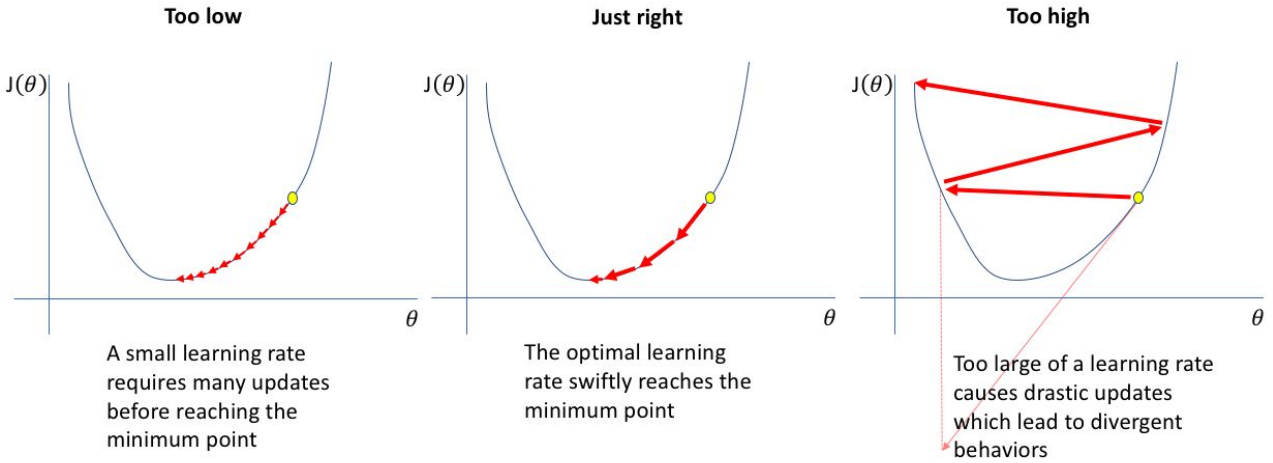
$$\frac{\partial E(\mathbf{a}, X, \mathbf{y})}{\partial \mathbf{a}} = 2(\mathbf{a}^T X^T - \mathbf{y}^T) \cdot X$$

Possible termination criteria:

- Fixed number of iterations
- Below some error threshold
- Small improvement across iterations

Basic gradient descent

Choosing the right step size is important.



Matrix decomposition

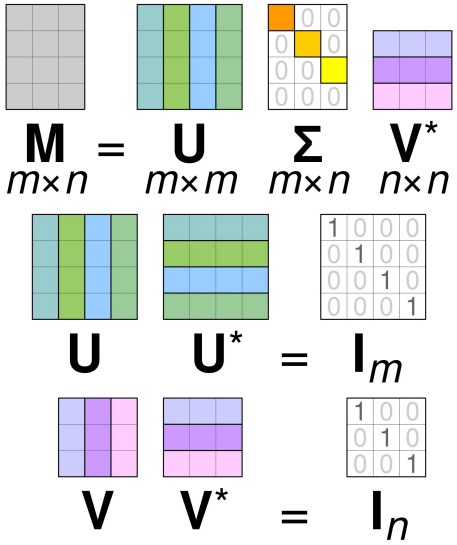
A set of techniques and algorithms for factoring matrices into multiplications of other matrices.

Applications:

- Improved efficiency in other computations (e.g. pseudo-inverse)
- Analyze matrix properties
- Data dimensionality reduction
- Latent feature extraction from data

SVD

SVD (singular value decomposition) is a specific technique of matrix decomposition, factoring a matrix into 3 matrices with specific properties (a generalization of eigenvalues).



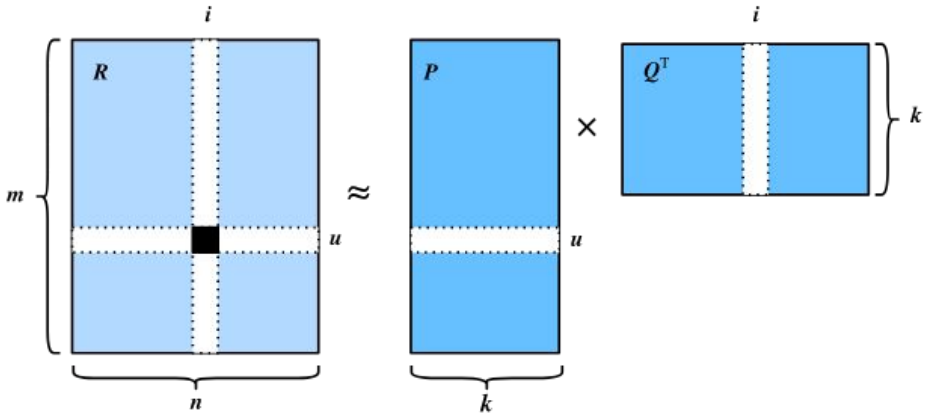
SVD

```
A = np.array([[1, 2], [3, 4], [5, 6]])

U, s, VT = np.linalg.svd(A)

print(A)
print('------')
print(U)
print('------')
print(s)
print('------')
print(VT.T)
```

```
[[1 2]
 [3 4]
 [5 6]]
------
[[-0.2298477   0.88346102  0.40824829]
 [-0.52474482  0.24078249 -0.81649658]
 [-0.81964194 -0.40189603  0.40824829]]
------
[9.52551809 0.51430058]
------
[[-0.61962948 -0.78489445]
 [-0.78489445  0.61962948]]
```

Netflix Prize challenge

Idea: use matrix decomposition in recommender systems, to jointly extract latent features of both users and items. This in turn can be used to predict specific users' interest in specific items.



References

- https://arxiv.org/abs/1802.01528 - the basis for most of the matrix calculus part of the lesson
- https://en.wikipedia.org/wiki/Gradient_descent
- https://en.wikipedia.org/wiki/Singular_value_decomposition
- https://machinelearningmastery.com/singular-value-decomposition-for-machine-learning
- https://sifter.org/~simon/journal/20061211.html
- https://en.wikipedia.org/wiki/Matrix_factorization_(recommender_systems)