DATA221 Intro Machine Learning
05 linear algebra

William Trimble
Spring 2023

THE UNIVERSITY OF CHICAGO

Why do we need linear algebra?

- So… X and Y are potentially large objects; vectors of thousands or millions of elements are possible.

- We’re going to need to perform operations that work and generalize to large datasets.

- So-called vectorized code (built into numpy) can efficiently perform operations on all the numbers.

The bare minimum of linear algebra…

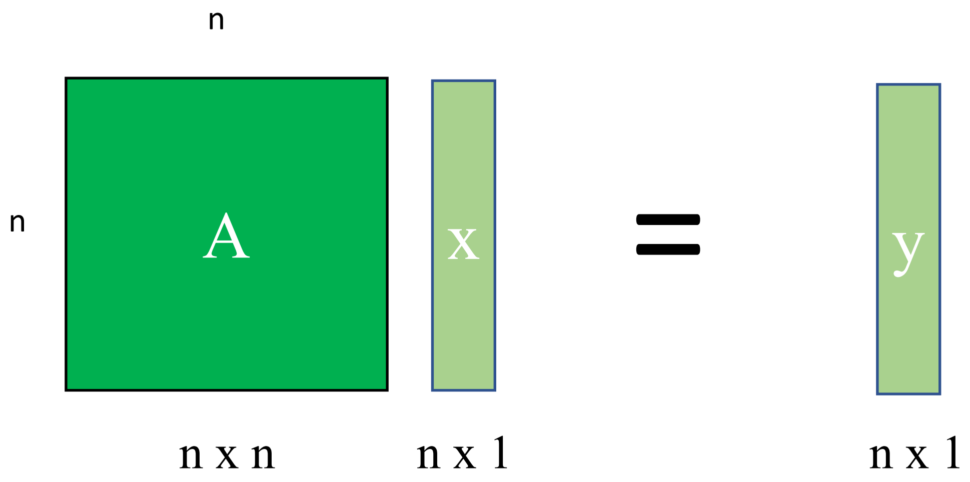
$$\mathbf{A}\,\mathbf{x} = \mathbf{y}$$

Square matrix A

vector x

vector y

square
matrix



The bare minimum of linear algebra…

Square matrix A

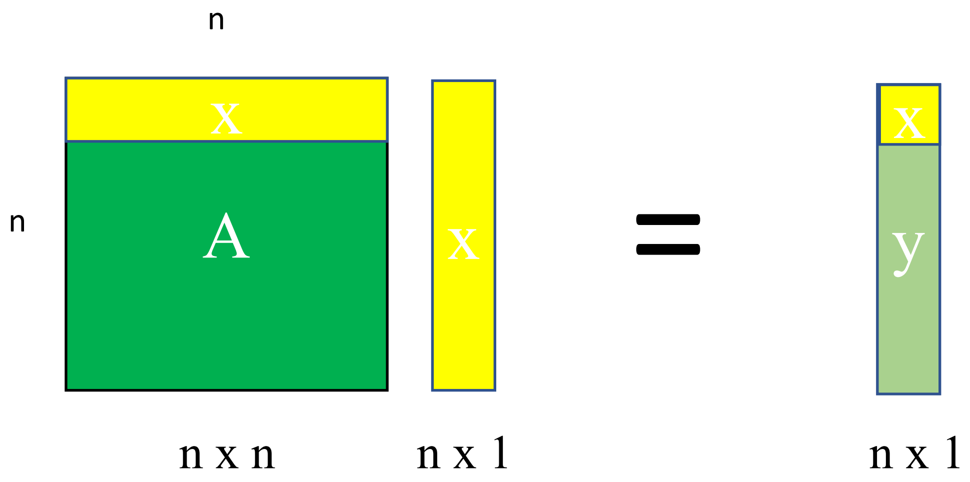
vector x

vector y

$$\mathbf{A}\,\mathbf{x} = \mathbf{y}$$

$$\Sigma A_{ij}\, x_j = y_i$$

square matrix



Definition of eigenvectors and eigenvalues

eigenector $\mathbf{e}_i$

eigenvalue λ_i

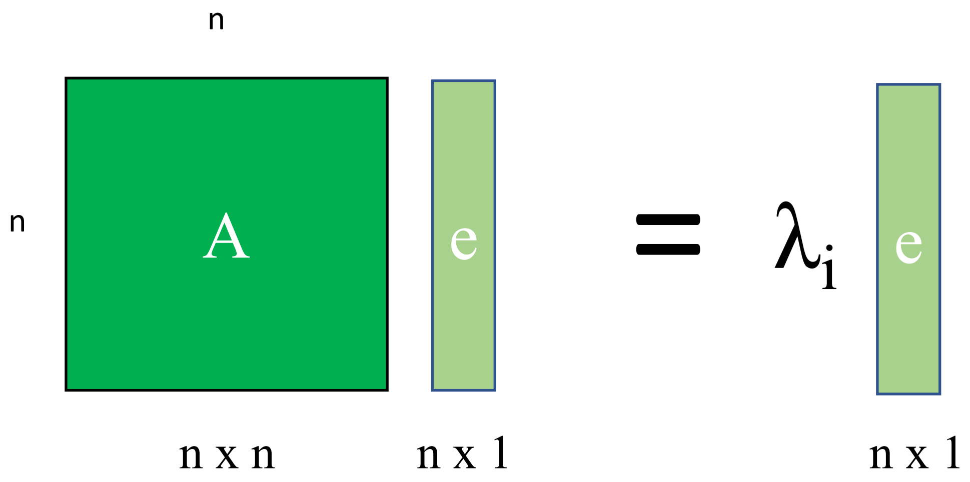
square matrix
n x n

ith eigenvalue

ith eigenvector
n x 1

$$\mathbf{A}\,\mathbf{e}_i = \lambda_i\,\mathbf{e}_i$$

square
matrix



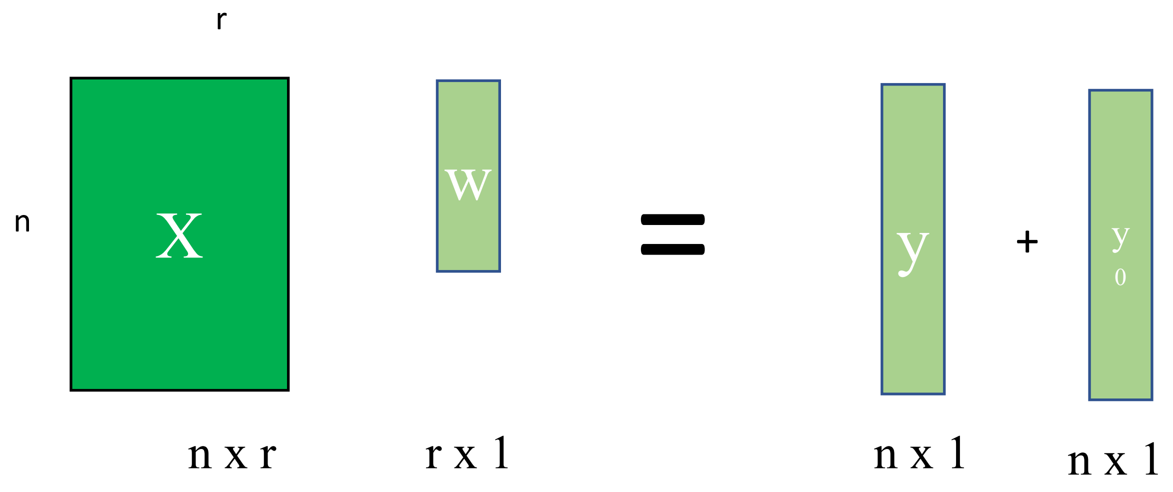
Rectangular matrix multiplication…

rectangle matrix
n x r

weights
r x 1

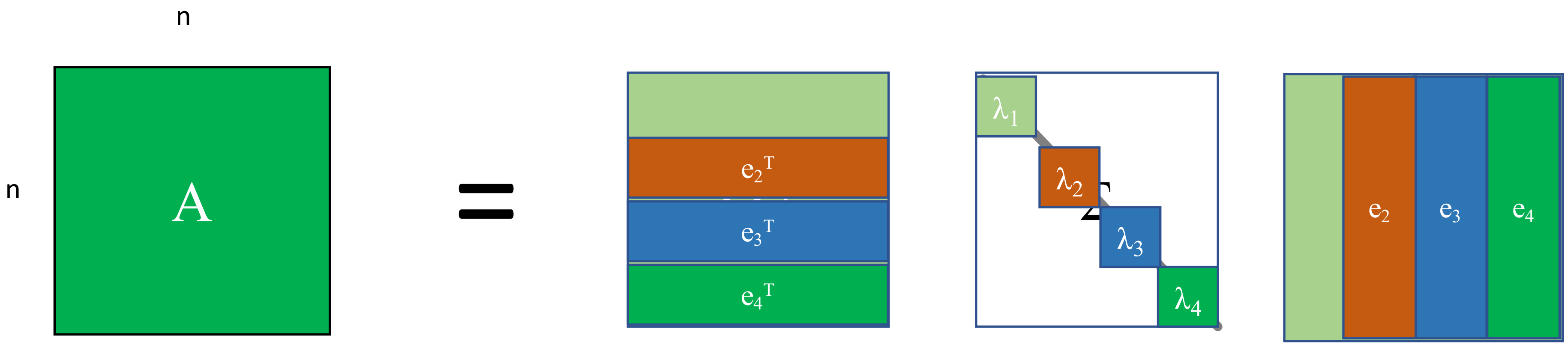
linear product / model
n x 1

$$\mathbf{X}\,\mathbf{w} = \mathbf{y} + \mathbf{y_0}$$



Eigenvalue decomposition

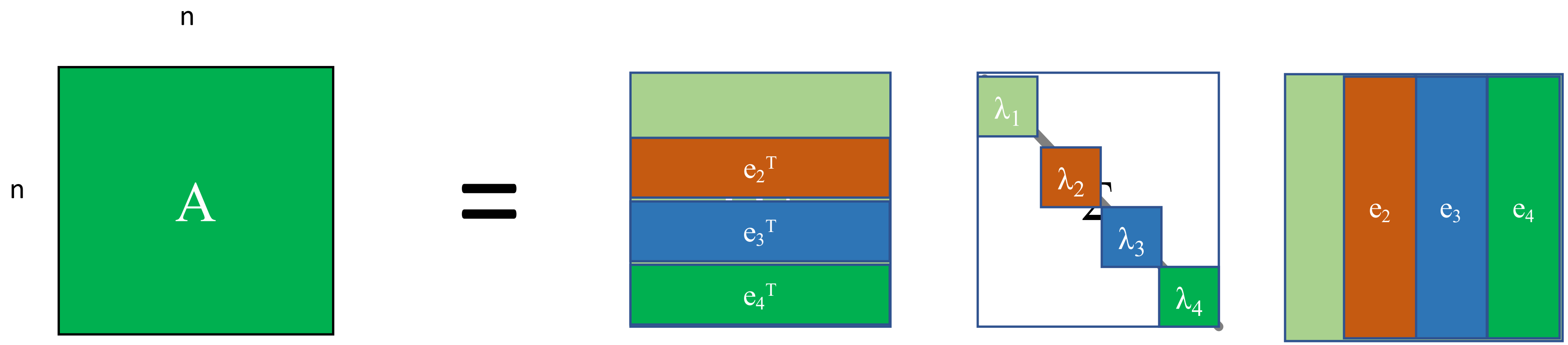
$$\mathbf{A} = \mathrm{Sum}_i \ \ \mathbf{e}_i^{T} \lambda_i \, \mathbf{e}_i$$



square matrix	eigenvectors	eigenvalues	eigenvectors
	(orthogonal)	(diagonal)	(orthogonal)
A	Q^{-1}	Σ	Q

Eigenvalue decomposition

$$\mathbf{A} = \mathrm{Sum}_i \ \mathbf{e}_i^T \lambda_i \mathbf{e}_i$$

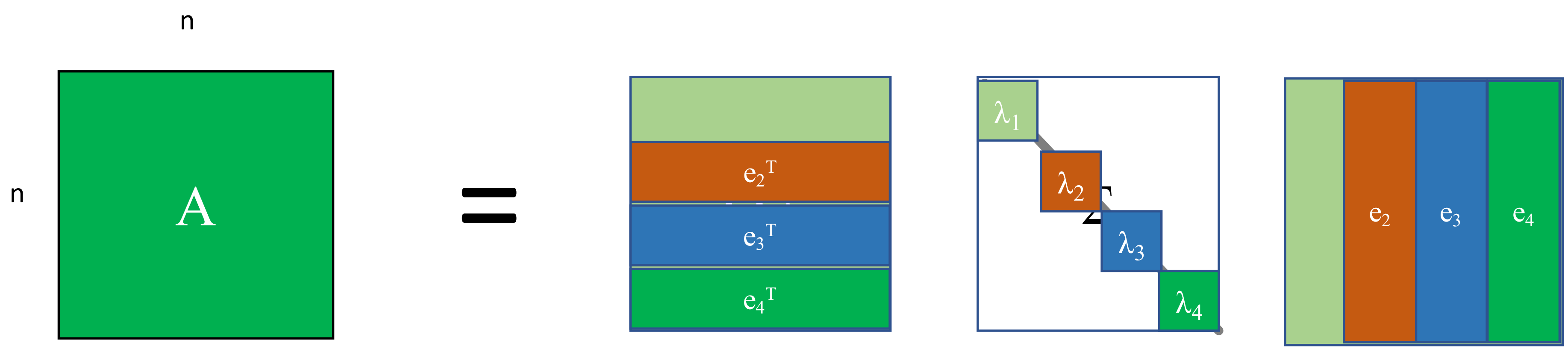


Why does the matrix have to be square?

A $\quad Q^{-1} \quad \Sigma \quad Q$

Eigenvalue decomposition

$$\mathbf{A} = \mathrm{Sum}_i \ \ \mathbf{e}_i^T \lambda_i \mathbf{e}_i$$

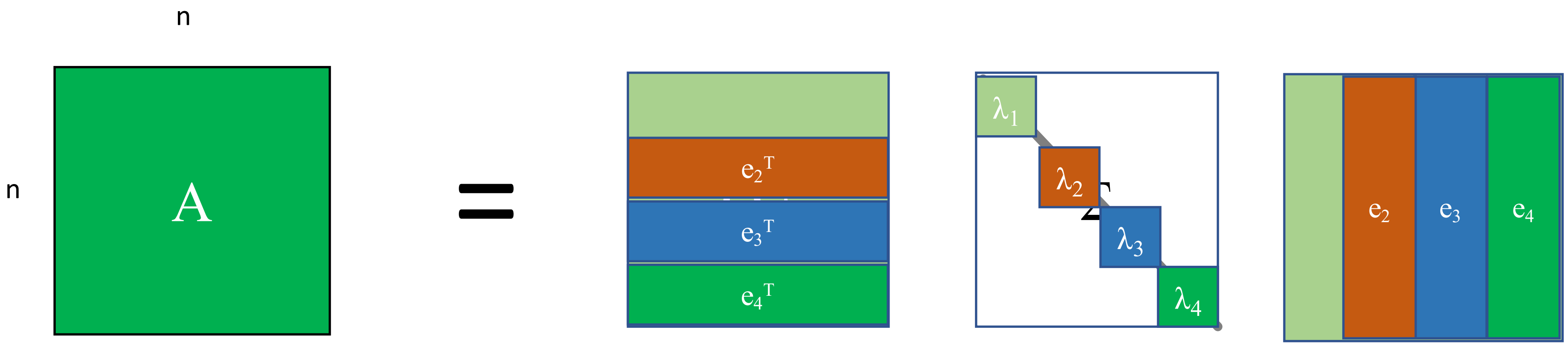


What happens if some of the eigenvalues are zero?

A $\qquad$ Q^{-1} $\qquad$ Σ $\qquad$ Q

Eigenvalue decomposition

$$\mathbf{A} = \mathrm{Sum}_i \ \ \mathbf{e}_i^T \lambda_i \, \mathbf{e}_i$$

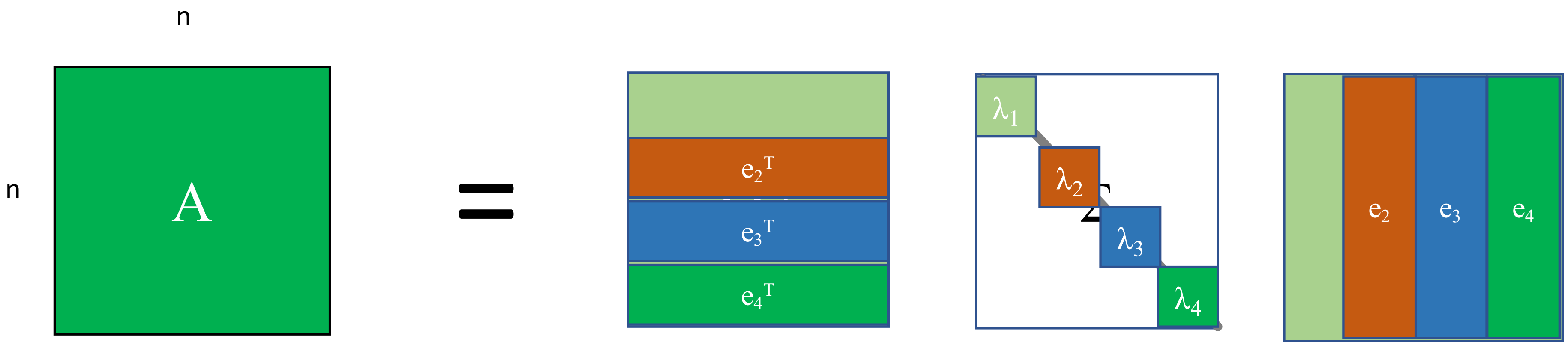


What happens when some of the eigenvalues are the same?

A $\quad$ Q^{-1} $\quad$ Σ $\quad$ Q

Eigenvalue decomposition

$$\mathbf{A} = \mathrm{Sum}_i \ \mathbf{e}_i^T \lambda_i \mathbf{e}_i$$



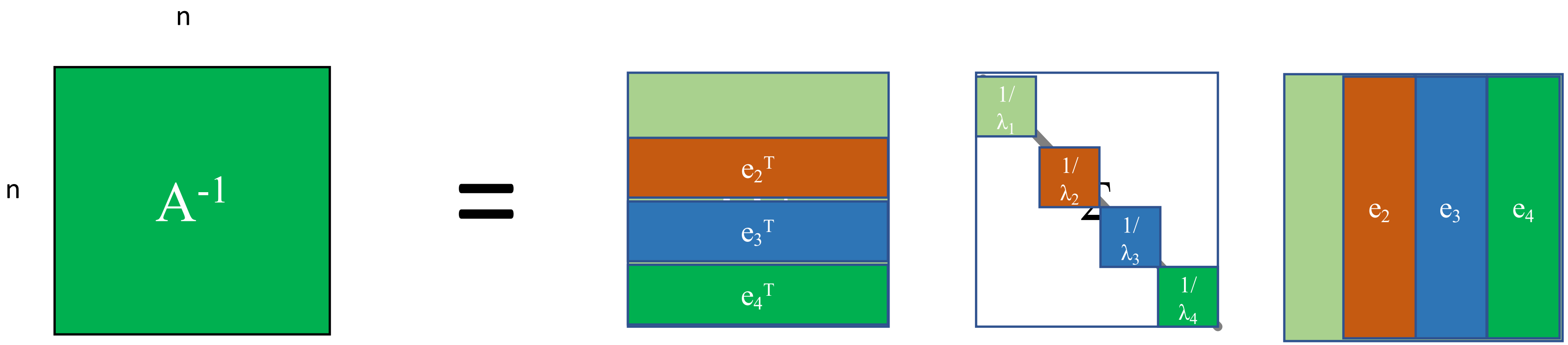
What happens when they aren't zero but are small?

(orthogonal) (diagonal) (orthogonal)

A $\quad Q^{-1} \quad \Sigma \quad Q$

Eigenvalue decomposition of the inverse

$$\mathbf{A}^{-1} = \mathrm{Sum}_i \ \mathbf{e}_i^T \lambda_i^{-1} \mathbf{e}_i$$



square matrix

A

eigenvectors

(orthogonal)

Q^{-1}

inverse of eigenvalues

(diagonal)

Σ^{-1}

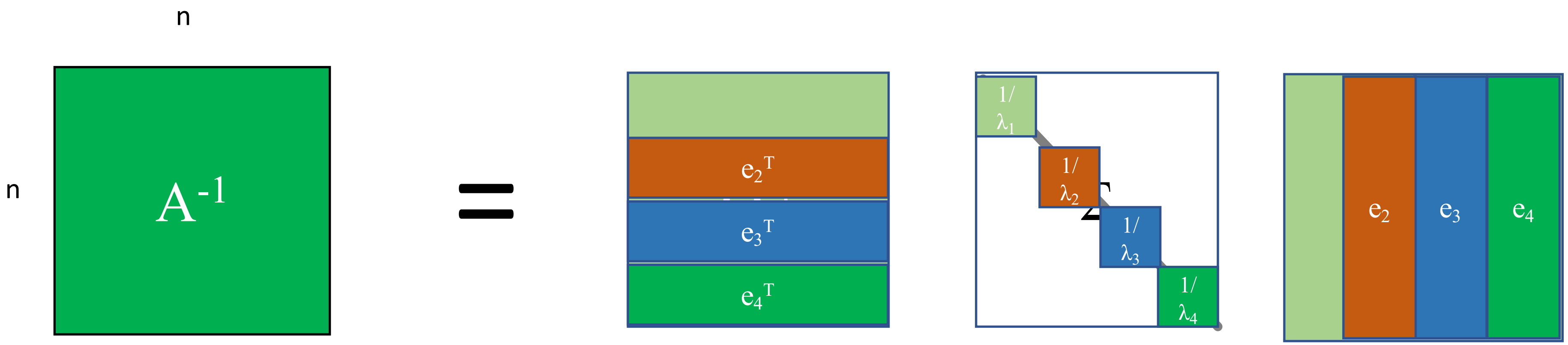
eigenvectors

(orthogonal)

Q

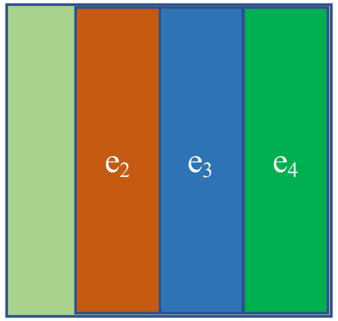
Eigenvalue decomposition of the inverse

$$\mathbf{A}^{-1} = \mathrm{Sum}_i\ \ \mathbf{e}_i^{T}\,\lambda_i^{-1}\mathbf{e}_i$$



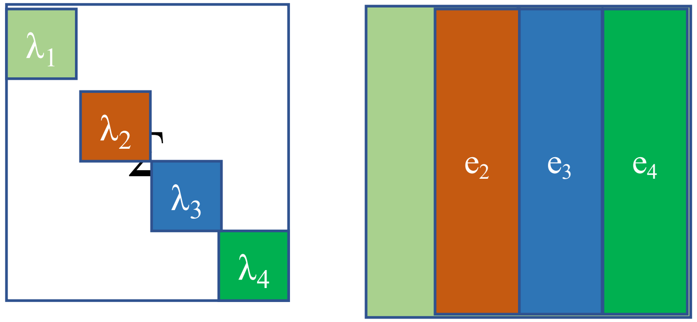
square matrix	eigenvectors (orthogonal)	inverse of eigenvalues (diagonal)	eigenvectors (orthogonal)
A	Q^{-1}	Σ^{-1}	Q

Why eigenvalue decomposition?



- Orthonormal basis is often convenient
- Orthonormal basis has a geometric interpretation; the directions the eigenvalues represent can help with gradient descent
- Permits truncation by importance; small-eigenvalue eigenvectors contribute less to the final product; we can store (and calculate) instead of n x m can store only n x p and p x n.
- How much memory does your computer have?
  - How large a square matrix can you hold in memory?
  - How large a square matrix will fit on your hard drive?

numpy.linalg.eig



`linalg.eig(a)` [source]

Compute the eigenvalues and right eigenvectors of a square array.

Parameters: **a** : *(..., M, M) array*

Matrices for which the eigenvalues and right eigenvectors will be computed

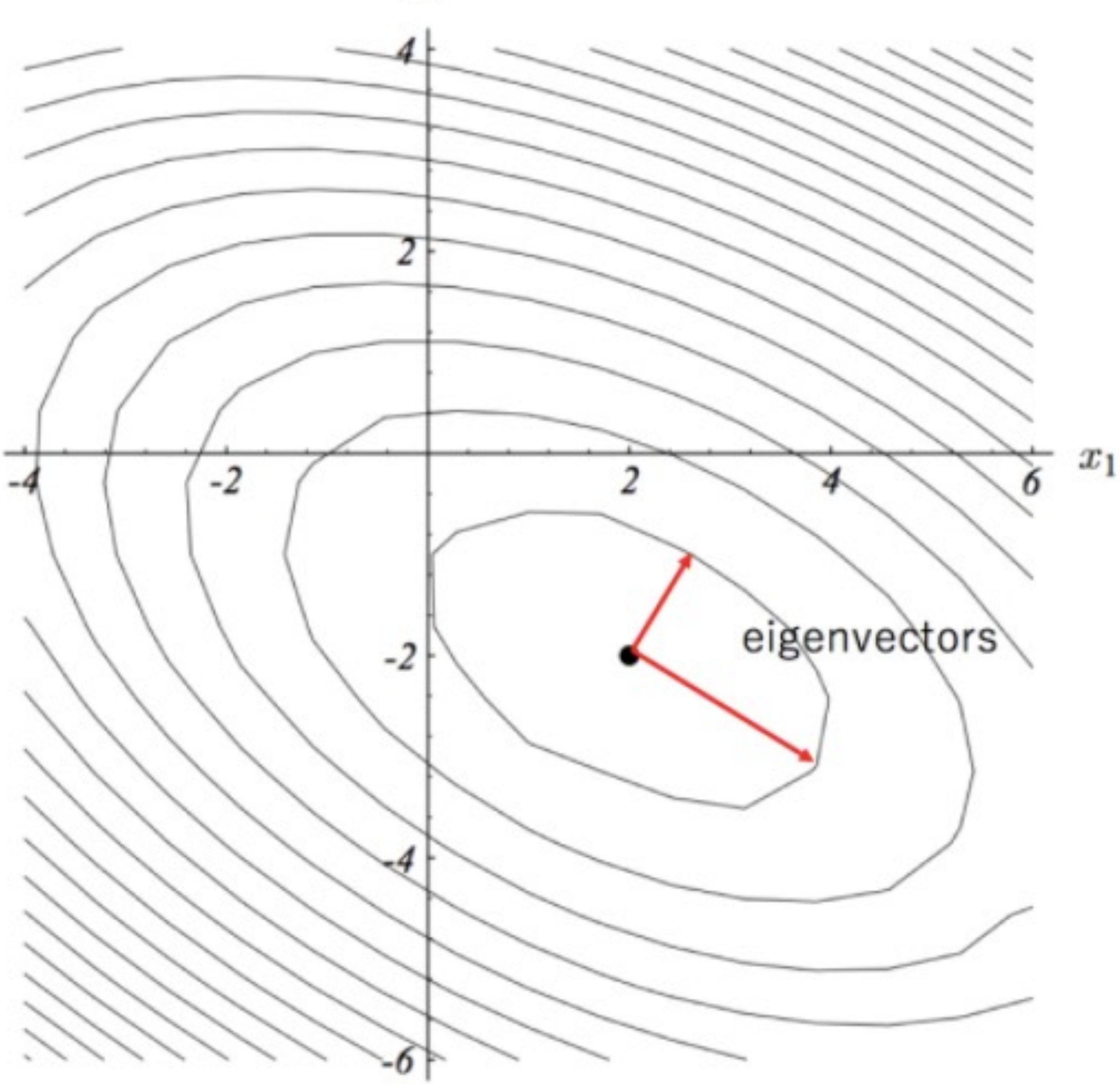
Returns: **w** : *(..., M) array*

The eigenvalues, each repeated according to its

v : *(..., M, M) array*

The normalized (unit “length”) eigenvectors, such that

For covariance matrices and second-derivative matrices, there is a geometric interpretation.



If you can recall Taylor expansion…

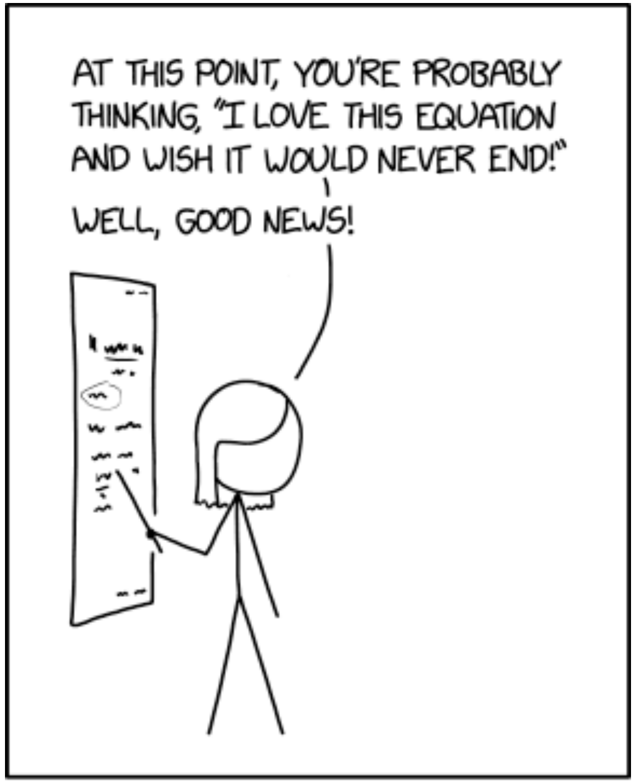
$$y = f(x + \Delta x) = f(x_0) + \frac{df}{dx}(x\text{-}x_0)$$

If you can recall Taylor expansion…

$$y = f(x + \Delta x) = f(x_0) + \frac{df}{dx}(x\text{-}x_0) + \frac{1}{2}\frac{d^2f}{dx^2}(x\text{-}x_0)^2$$

If you can recall Taylor expansion…

$$y = f(x + \Delta x) = f(x_0) + \frac{df}{dx}(x\text{-}x_0) + \frac{1}{2}\frac{d^2 f}{dx^2}(x\text{-}x_0)^2$$



TAYLOR SERIES EXPANSION IS THE WORST.

The multidimensional equivalent

$$y = f(\mathrm{x} + \Delta \mathrm{x}) \ = \ f(\mathrm{x}_0) + \ \frac{df}{dx}\,(\mathrm{x}\text{-}\mathrm{x}_0) + \frac{1}{2}\,\frac{d^2 f}{dx^2}(\mathrm{x}\text{-}\mathrm{x}_0)^2$$

$$y = f(\mathbf{x} + \Delta\mathbf{x}) \approx f(\mathbf{x}) + \nabla f(\mathbf{x})^{\mathrm{T}} \Delta\mathbf{x} + \frac{1}{2}\,\Delta\mathbf{x}^{\mathrm{T}}\,\mathbf{H}(\mathbf{x})\,\Delta\mathbf{x}$$

The multidimensional equivalent

$$y = f(\mathrm{x} + \Delta \mathrm{x}) \;=\; f(\mathrm{x}_0) + \frac{df}{dx}(\mathrm{x}\text{-}\mathrm{x}_0) + \frac{1}{2}\frac{d^2 f}{dx^2}(\mathrm{x}\text{-}\mathrm{x}_0)^2$$

$$y = f(\mathbf{x} + \Delta \mathbf{x}) \approx f(\mathbf{x}) + \nabla f(\mathbf{x})^{\mathrm{T}} \Delta \mathbf{x} + \frac{1}{2} \Delta \mathbf{x}^{\mathrm{T}} \mathbf{H}(\mathbf{x}) \Delta \mathbf{x}$$

function value at x

1^{st} derivative

2^{nd} derivative matrix

Hessian matrix

Take a scalar-valued function $f(x_1, x_2, x_3\ldots x_n)$.

The second derivative of this function is a n x n matrix:

$$\mathbf{H}_f = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1\,\partial x_n} \\[2.2ex] \dfrac{\partial^2 f}{\partial x_2\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2\,\partial x_n} \\[2.2ex] \vdots & \vdots & \ddots & \vdots \\[2.2ex] \dfrac{\partial^2 f}{\partial x_n\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix},$$

Diagonals are the 2nd derivatives in the directions of the basis vectors $e_1, e_2\ldots$

Off-diagonal terms measure essentially the same thing in different directions.

Matrix is symmetric, so eigenvalues are real. I can use eigenvalue decomposition to understand geometry of the function's curvature.

Hessian matrix

Take a scalar-valued function $f(x_1, x_2, x_3 \ldots x_n)$.

The second derivative of this function is a n x n matrix:

$$\mathbf{H}_f = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1\,\partial x_n} \\ \dfrac{\partial^2 f}{\partial x_2\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2\,\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix},$$

The eigenvalue decomposition of this matrix will tell you

the direction (in R^n) of maximum 2nd derivative curvature

the maximum curvature

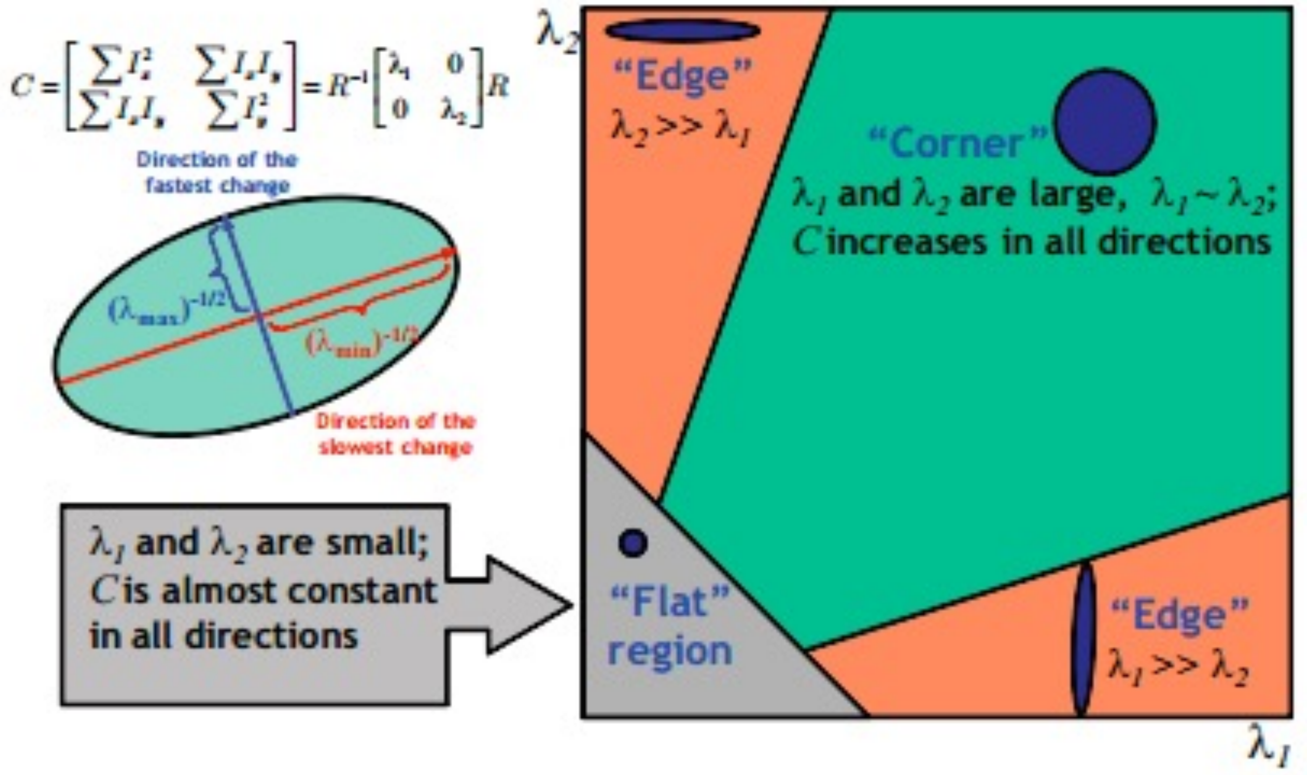
the direction of second-maximum curvature…

the direction of third-maximum curvature…

Hessian matrix

$f(x_1, x_2, x_3 \ldots x_n) \ldots$

$$\mathbf{H}_f = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1\,\partial x_n} \\[2.2ex] \dfrac{\partial^2 f}{\partial x_2\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2\,\partial x_n} \\[2.2ex] \vdots & \vdots & \ddots & \vdots \\[2.2ex] \dfrac{\partial^2 f}{\partial x_n\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix},$$

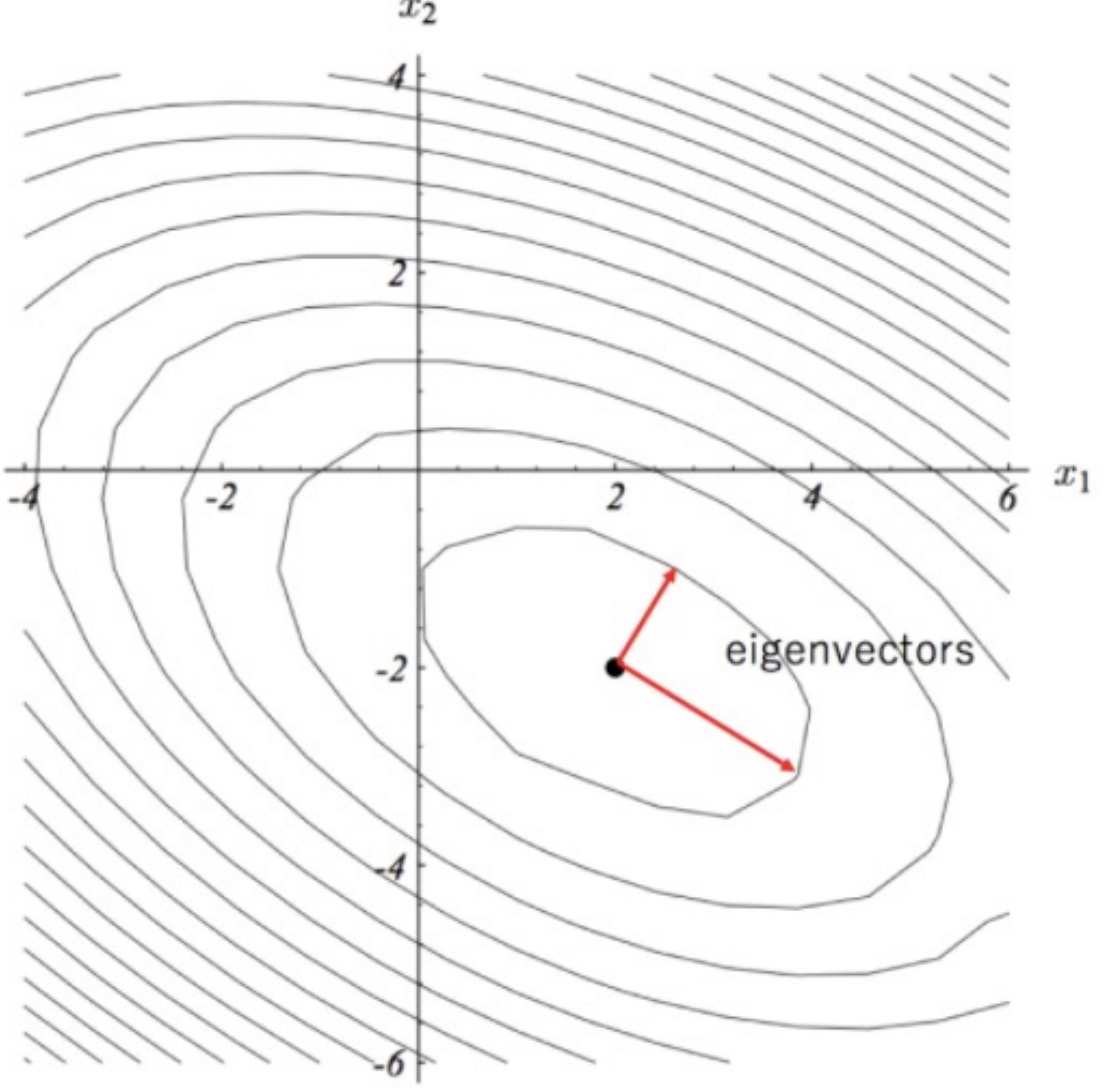


https://stackoverflow.com/questions/22378360/hessian-matrix-of-the-image

Hessian matrix

$f(x_1, x_2, x_3 \ldots x_n) \ldots$

$$\mathbf{H}_f = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1\,\partial x_n} \\ \dfrac{\partial^2 f}{\partial x_2\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2\,\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n\,\partial x_1} & \dfrac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix},$$



https://stackoverflow.com/questions/22378360/hessian-matrix-of-the-image