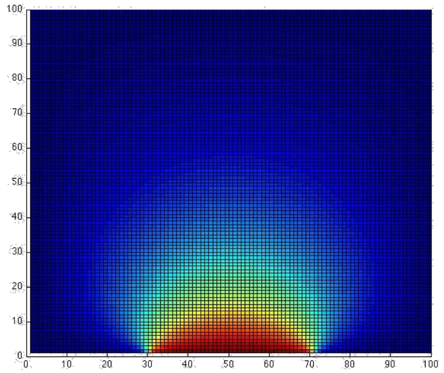

Lecture 05:

Eigenvalues & Eigenvectors Cholesky's Method

CS 111: Intro to Computational Science

Spring 2023

Ziad Matni, Ph.D.

Dept. of Computer Science, UCSB

Administrative

- Old homework to turn in!
- New homework to begin!

- Reminder: Quiz 2 on Wednesday

Eigenvalues!

*Eigenvalues and eigenvectors have a wide range of applications, for example in **computer graphics**, **facial recognition**, and **matrix diagonalization**.*

*Consider: **A** is a matrix, **x** is a vector*

When **Ax = λx** , where λ is a scalar,

We call λ an <u>*eigenvalue*</u> of **A** (there can be more than 1 of these)

and the vector **x** is an <u>*eigenvector*</u> of **A** (there can be as many of these as λs)

Also means: the **Ax** vector is <u>parallel</u> to **x** and "stretched" by a factor of λ

<u>Properties of Eigenvalues (that can help you calculate λ)</u>

1. **Sum of all λ equals the sum of all the *diagonal values in A – this is a.k.a.* trace(A)**
2. **Product of all λ equals det(A)**
3. **det(A - λ I) = 0**

Eigenvalues and Eigenvectors

With $\mathbf{A}\,\mathbf{x} = \lambda\,\mathbf{x}$ We call λ the eigenvalue and **x** is the eigenvector

and $\lambda_1, \lambda_2, \ldots \lambda_n \in \mathbb{C}$

<u>Related theorem 1:</u>

- **A** and $\mathbf{A}^T$ have the same *eigenvalues*, but usually <u>different</u> *eigenvectors*

- And by the way… Didja know that?
  - Given *any* matrix **A**, $\mathbf{B} = \mathbf{A} + \mathbf{A}^T$ is <u>always symmetrical</u>!!?! Can we prove this?
  - This is a great way to construct a symmetrical matrix from scratch!

<u>Related theorem 2:</u>

- If matrix **A** is real (that is all values $a_{ij} \in \mathbb{R}$) and it's symmetrical, then all its *eigenvalues* $\in \mathbb{R}$ *too*

Example

Let $A = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$

So, **trace(A)** = sum of all λ, so it's = 1 + 2 = **3**

And, **det(A)** = product of all λ , so it's = 1x2 = **2**

So what are the values λ?

$\lambda_1 + \lambda_2 = 3$
$\lambda_1 . \lambda_2 = 2$

$\Rightarrow \lambda_1 = 3 - \lambda_2$
$\Rightarrow (3 - \lambda_2) . \lambda_2 = 2$
$\Rightarrow \lambda_2^2 - 3\lambda_2 + 2 = 0$
$\Rightarrow (\lambda_2 - 1)(\lambda_2 - 2) = 0$
$\Rightarrow$ **$\lambda = (1, 2)$**

Class Exercise

Let $A = \begin{pmatrix} -1 & 4 \\ 5 & -2 \end{pmatrix}$

Find the values of λ

Eigenvalues and Eigenvectors

- `numpy.linalg.eig(A)` returns 2 vectors **d** and **V**:
  - **d** = vector with all of A's eigenvalues
  - **V** = the eigenvectors presented as a matrix

- `numpy.linalg.eigh(A)` does the same thing *only faster*
  - Uses a different algorithm than `.eig()`
  - BUT should **only** be used if you know **A** is symmetrical
  - FYI: the **h** is for "Hermitian"

Eigenvalues and Eigenvectors

- For a *n-by-n* square matrix: $(\mathbf{A} - \lambda\mathrm{I})\boldsymbol{x} = 0$ is *always true*
- It implies that $(\mathbf{A} - \lambda\mathrm{I})$ is **singular** meaning that: $\det(\mathbf{A} - \lambda\mathrm{I}) = 0$
    - We're ignoring the case when **x** is a zero vector – trivial issue.
- This is mainly how we solve for λ and **x**
- SO: Let's consider the matrix Λ:

    n-by-n diagonal matrix with eigenvalues λ_j as elements

$$\Lambda = \begin{pmatrix} \lambda_0 & 0 & 0 & \dots \\ 0 & \lambda_1 & 0 & \dots \\ 0 & 0 & \lambda_2 & \dots \\ \dots & & & \end{pmatrix}$$

- ..and also consider the matrix **X**:

    n-by-n set of corresponding eigenvectors $\mathbf{x}_j$ for each λ_j

$$X = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots \\ X_0 & X_1 & X_2 & \dots \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

$$\Lambda = \begin{pmatrix} \lambda_0 & 0 & 0 & \dots \\ 0 & \lambda_1 & 0 & \dots \\ 0 & 0 & \lambda_2 & \dots \\ \dots & & & \end{pmatrix}$$

$$X = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots \\ X_0 & X_1 & X_2 & \dots \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

Eigenvalues and Eigenvectors

Continued…

- Therefore: Since **Ax = λx** then you can show that: **AX = XΛ**

- If we multiply both sides by **X**$^{-1}$ (<u>if and only if</u> it exists!), then:

$$AXX^{-1} = X\Lambda X^{-1} \quad \textit{...or...}$$

$$\boxed{A = X\Lambda X^{-1}}$$

- *This is called the "**eigendecomposition of a matrix**" (another factorization technique)*
  - *We will relate this to another technique later in the course (called SVD)*

Symmetrical Positive Definite (SPD) Matrices

SPD = Symmetrical Positive Definite

- **SPD Matrices** have similar properties as “positive numbers” in scalar math
  - When you multiply an SPD Matrix with any vector, its direction stays “similar” to the vector

- **SPDs** show up a lot in physical world measurements that go into statistical models, control system designs, heat conductivity designs, etc…

- There are many good algorithms (**fast**, **numerically stable**) that work better for an **SPD** matrix, such as **Cholesky factorization**.
  - Instead of using LU factorization

Characteristics of SPD Matrices

If matrix A is SPD, then that means…

- $\mathbf{x}^T\mathbf{A}\mathbf{x} > 0$ for every non-zero $\mathbf{x}$
  - You can only use this in a proof-by-induction, so it's not always handy
- Mathematically also means that **all** its eigenvalues are **> 0**
  - Easier to calculate than that first rule
- A is symmetrical, i.e. $A = A^T$

Exercise: What are the Eigenvalues of…

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

`λ1 + λ2 = sum(diagonals) = 2 + 2 = 4` (*again, we call this the "**trace**" of matrix A*)

`λ1.λ2 = det(A) = 2x2 – 1x1 = 3`

So, `λ1 = 4–λ2 and (4–λ2).λ2 = 3`

And after some math (quadratics!! yay!!)… **`λ1 = 1`** & **`λ2 = 3`**

Is A an SPD matrix?

Python Demonstration

Cholesky Factorization

- For certain cases, we might have a matrix **A** that is a:
  - Symmetric square matrix
  - and is *positive definite* (again, meaning: all the ***eigenvalues*** of **A** are positive)

- André-Louis Cholesky was an artillery officer in the French army who came up with this technique to help calculate bomb trajectories!
  - See how SPD matrices come up a lot in real physical models?...

- It turns out that, for an SPD **A** matrix, we can factor **A** into $\mathbf{RR^T}$
- It means that you only need to calculate one factor matrix here: **R**

Cholesky Factorization

- Cholesky factorization is a particular form where **R** is a **lower triangular with positive diagonals**

Lower Triangular L

$$\begin{bmatrix} A_{00} & A_{01} & A_{02} \\ A_{10} & A_{11} & A_{12} \\ A_{20} & A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_{00} & 0 & 0 \\ L_{10} & L_{11} & 0 \\ L_{20} & L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} L_{00} & L_{10} & L_{20} \\ 0 & L_{11} & L_{21} \\ 0 & 0 & L_{22} \end{bmatrix}$$

Transpose of L

Easier
(almost 2x faster) to calculate than
***A = LU** factorization*

Easy Cholesky Calculation Example

- Factor matrix $\mathbf{A} = \mathbf{RR}^\mathbf{T}$ (that, is find matrix **R**)
- Example:

$$\begin{bmatrix} 4 & -1 \\ -1 & 3 \end{bmatrix} = \begin{bmatrix} a & 0 \\ b & c \end{bmatrix} \begin{bmatrix} a & b \\ 0 & c \end{bmatrix} = \begin{bmatrix} a^2 & ab \\ ab & b^2+c^2 \end{bmatrix}$$

So:

$$a^2 = 4 \rightarrow a = 2$$

$$ab = -1 \rightarrow b = -0.5$$

$$b^2+c^2 = 3 \rightarrow c = \text{sqrt}(2.75) = 1.658$$

$$R = \begin{bmatrix} 2 & 0 \\ -0.5 & 1.658 \end{bmatrix}$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 3 \end{bmatrix}$$

Cholesky Example Continued…

- Factor matrix **A = RR**^{T} (that, is find **R**)
- So:

$$\begin{bmatrix} 4 & -1 \\ -1 & 3 \end{bmatrix} = \overset{R}{\begin{bmatrix} 2 & 0 \\ -0.5 & 1.658 \end{bmatrix}} \overset{R^T}{\begin{bmatrix} 2 & -0.5 \\ 0 & 1.658 \end{bmatrix}}$$

Check work?

$$\begin{bmatrix} 4 & -1 \\ -1 & \underbrace{-0.5^2 + 1.658^2}_{2.99999} \end{bmatrix}$$

Python Demonstration

```
R = npla.cholesky(A)
```

Your TO DOs!

- Turn in your homework by today
- Quiz#2 Wednesday
- Lab Thursday!

</LECTURE>