

Note

Disini tempat saya menguji pemahaman saya tanpa bantuan apapun, untuk mengukur kemampuan saya. Jawaban saya murni

Pertanyaan #1: Tentang Aljabar Linear (Vektor & Transformasi)

Bayangkan kamu bertemu dengan seorang teman dari jurusan non-teknik. Dia melihatmu mengerjakan soal aljabar linear dan bertanya:

Example scenario

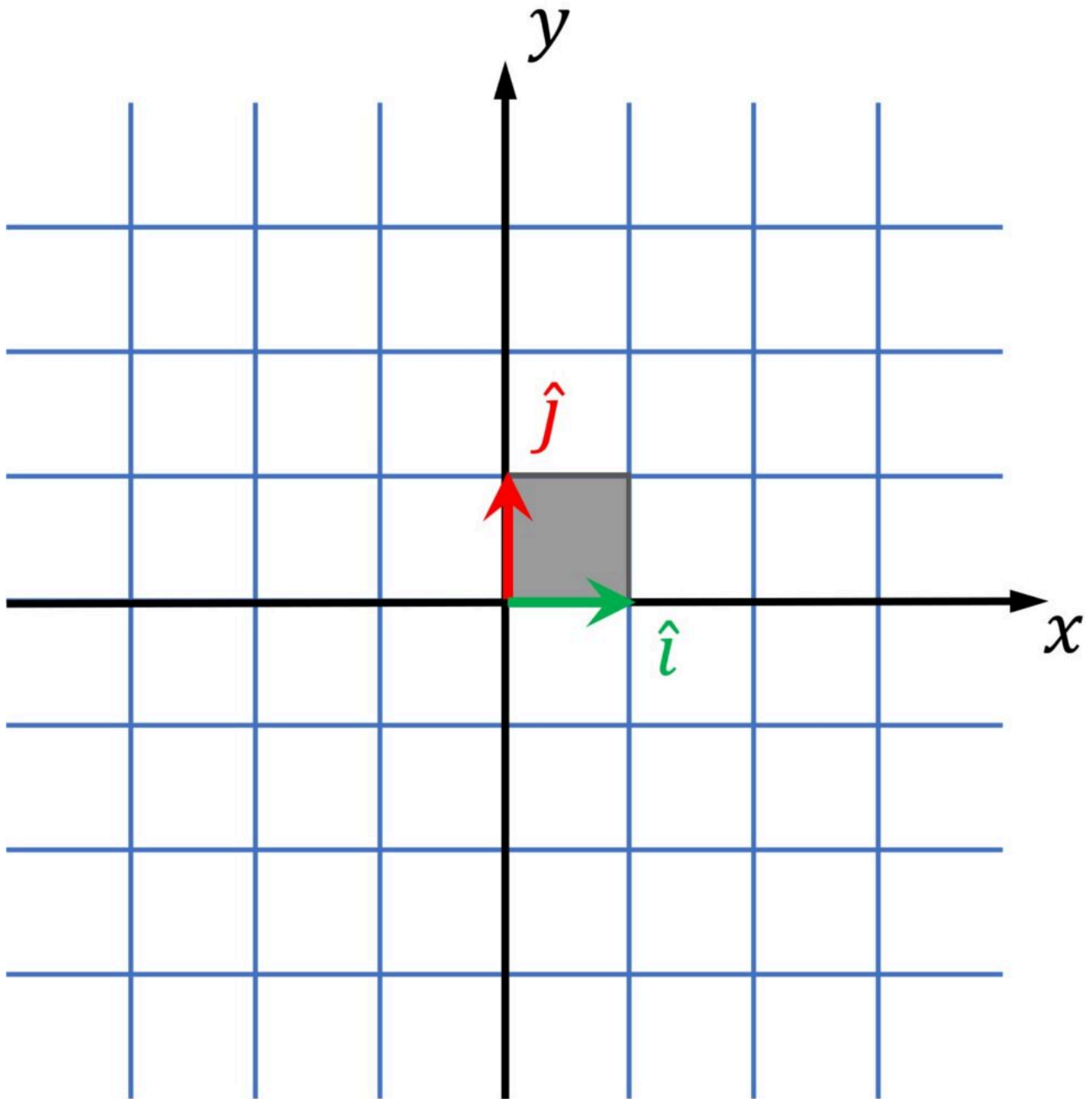
"Aku lihat kamu mengalikan sebuah 'matriks' $\begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}$ dengan sebuah 'vektor' $\begin{bmatrix} 1 & 1 \end{bmatrix}$. Aku nggak ngerti sama sekali. Apa sih yang sebenarnya sedang kamu lakukan? Kenapa kotak angka dikalikan dengan daftar angka? Apa artinya itu secara visual?"

Tugasmu:

Jelaskan kepadanya **dengan analogi atau visualisasi sederhana** (tanpa harus menghitung hasilnya) apa arti dari perkalian matriks-vektor itu. Hubungkan dengan ide-ide yang sudah kita pelajari (misal: dari 3Blue1Brown atau Coursera).

Jawaban :

Apa sih arti dari matriks itu? nah matriks itu sendiri artinyaaa adalah tranformasi linear. Hmm mungkin masih belum kebayang nih? Oke coba bayangkan kamu membuat sebuah koordinat 2 dimensi, seperti pada biasanyaa yang sering kali dibahas di waktu kuliah yaitu ada sumbu x dan y. Disini bedanyaa, aku mau kamu bener benerr masuk ke 2 dimensi itu, dimana maksud saya adalah bayangin seperti ada grid dengan panjang an ukuran yang sama ("Evenly Spaced"). Sehingga kayak ada banyak kotak kotak persegi, seperti gambar dibawah ini :



Oke sekarang kamu sudah mendapatkan visualisasinyaa gimanaaa yaaa, nah apa sih maksud dari matriks :

$$\begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \text{?????}$$

Itu maksudnya adalah kamu seperti meregangkan, memutar, dan sebagainya dunia 2 dimensi mu. Dalam kasus ini gimana?? sekarang bayangin dulu enggak ada vektor 1,1 nya yaa. Fokus dengan $[2, 1]$, $[1, 3]$. Nah fokus ke kolom pertama yaitu $2, 1$, itu artinya, untuk vektor i dari yang awalnya $[1, 0]$ pergi ke titik $2, 1$. Bayangkan sekarang bentuk gridnya berubah. Nah begitu juga, pada kolom kedua, berarti vektor j pergi ke $1, 3$. Nah dia akan membentuk suatu grid yang berbeda. Akan tetapi dia memenuhi syarat yaitu evenly spaced dan sumbu titik pangkalnya tidak berubah. Sekarang kembali lagi ke awa, katakan ada vektor $1, 1$ di 2 dimensi mu, sekarang posisi dia pasti berubah kan, karena dipengaruhi oleh transformasi linear. Untuk hasilnya, kita bisa fokus pada dimana titik i dan j jatuh, i baru kali

dengan x dan j baru dikali dengan y . Dijumlahkan, maka ketemulah hasil akhirnya yaitu $[3,4]$.

Jadi, vektor $[1, 1]$ yang di dunia lama artinya '1 langkah ke kanan, 1 langkah ke atas', setelah 'dunianya' kita obrak-abrik dengan matriks itu, sekarang ia mendarat di lokasi baru, yaitu $[3, 4]$.

Pertanyaan #2: Tentang Kalkulus (Turunan & Integral)

Temanmu yang sama datang lagi. Kali ini dia melihat catatan kalkulusmu dan menunjuk ke dua ide yang berbeda:

1. Gambar **irisan-irisan tipis di bawah kurva** $y = x^2$ (seperti di Chapter 1 3b1b).
2. Gambar **persegi $x \times x$ yang ditambah irisan dx** (seperti di Chapter 3 3b1b).

Dia bingung dan bertanya:

☰ Example scenario

"Ini kok sama-sama 'irisan tipis' tapi kelihatannya beda ya? Yang satu kayaknya buat nyari luas di bawah kurva, yang satu lagi buat nyari turunan dari x^2 . Apa bedanya? Kenapa masalah yang satu (mencari Integral) malah butuh konsep turunan, dan masalah yang lain (mencari Turunan) pakai gambar persegi?"

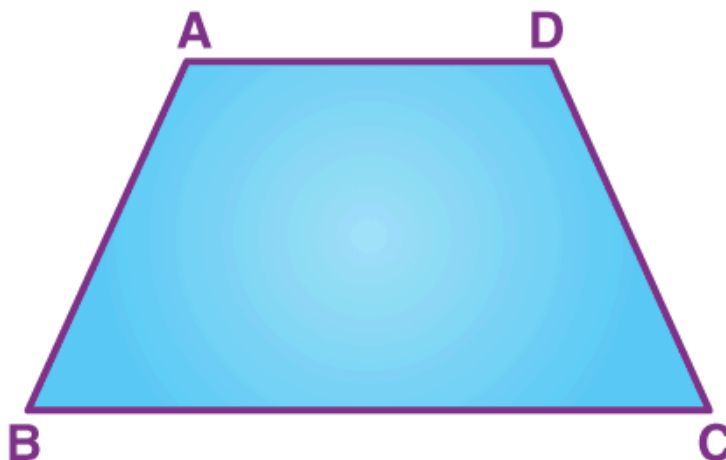
Tugasmu:

Jelaskan perbedaan fundamental antara kedua visualisasi "irisan tipis" tersebut. Jelaskan **tujuan** dari masing-masing visualisasi dan kenapa mereka saling berhubungan melalui Teorema Fundamental Kalkulus.

Jawaban:

Aku jujur kurang begitu paham dengan pertanyaanmu, jadi mungkin aku akan membawamu bareng bareng ke pemahaman paling dasar dulu yaa. Apa sih itu integral??

Kamu pernah kepikiran enggak, kenapa sih kok luas dari sebuah lingkaran itu sama dengan πr^2 ? Darimana sih rumus itu didapat? Well, aada banyak cara sih untuk menjelaskan ini didapat dari mana, ada yang pakai perumpamaan potongan pizza dan sebagainya. Nah sekarang kita pakai perumpamaan seperti ini : Bayangin kita memiliki sebuah lingkaran yaa, terus kita potong potong menjadi banyak potongan cincin. Nah cincin ini kalau kamu bayangin bentuknyaaa, akan berbentuk seperi trapezoid, lihat gambar di bawah :



Nah lihat yaa itu ada sisi atas dan ada sisi bawah yaitu AD dan BC ya kan. Nah sekarang kalau kamu lihat itu kan potongannyaa gede banget ya kan, sekarang coba bayangin kalau irisannyaa muakinnnnn tipisssssssss bangetttt. dia bakalan jadi bentuk apa? yak hampir seperti persegi panjang. dan makin tipisssssss lagi sampe mendekati nol, dia akan berbentuk persegi panjang ya kan? Nah anggap sekarang yaa, kamu bayangin itu adalah persegi panjang. Nah sisi AD dan BC itu panjang nya adalah keliling lingkaran yaitu $2\pi r$, lalu untuk tinggi dari persegi panjang yaitu si AB dan DC itu kan panjang bisa berubah berubah ya kan, tergantung se tipis dan se tebal apa sih, berarti kita anggap saja yaitu dr ($d = \text{difference}$). sehingga luasnya adalah $2\pi r dr$

Okee sekarang bayangkan nih irisannyaa buanyakk poll kita susun berdiri, nah akan membenrtuk suatu segitu siku siku. Dimana luasnya adalah $\frac{1}{2} 2\pi r r$, sehingga jadinya adalah πr^2

Nah berarti integral itu sendiri adalah jumlah luas dibawah grafik. Lihat simbolnya, layaknya sebuah s panjang (sum).

Bagaimana turunan? well turunan mungkin seperti yang kamu tau, Definisi formal turunan (derivatif) adalah

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

Penjelasan Rinci:

- **Konsep Dasar:** Turunan adalah perpanjangan dari konsep gradien garis singgung pada suatu titik di kurva. Bayangkan garis sekan (garis potong) yang menghubungkan dua titik pada kurva. Ketika jarak antara kedua titik tersebut menjadi sangat kecil (mendekati nol), garis sekan itu akan menjadi garis singgung, dan gradiennya adalah turunan.

- **Pentingnya Limit:** Konsep limit memastikan bahwa kita bisa mengukur perubahan pada satu titik tunggal, bukan hanya perubahan rata-rata pada sebuah interval.- **Kegunaan:** Turunan sangat penting untuk memahami kemiringan kurva, kecepatan, percepatan, optimasi (mencari nilai maksimum/minimum), dan banyak aplikasi lain di berbagai bidang ilmu

Nahh apa hubungannyaa antara integral dengan turunan? sekarang coba bayangin ada sebuah fungsi dengan $f(x) = x^2$. Kebayang kan gambarnya gimana? terus kalau kamu disuruh mencari luas dibawah kurva tersebut, pastinya kesusahan kan? (meskipun ada rumus dan sebagainyaa, tapi pov kita saat ini anggap kita enggak tau apa apa yaa).

Sekarang coba bayangin yaaa, kita ketahui luas dibawah kurva x^2 yaitu $A(x)$, yaitu luas dari titik nol hingga x . Nah apabila kita geser sedikittt ke kanan untuk x nyaa, maka akan ada perubahan untuk luasnyaa kan? menjadi dA (difference A), dimana kita bisa menganggap luas perubahan nya mendekati nol dan hampir berbentuk persegi panjang, sehingga $dA = dx$ (panjang) dikali x^2 (tinggi), yang dimana dengan aturan aljabar, maka $dA/dx = x^2$. Apa maksudnyaaa??? persamaan dA/dx itu adalah turunan (diferensial), yang dimanaaa berarti untuk mencari luas (integral) dibawah kurva itu sama mencari anti turunan dari x^2 .

Itulah teorema fundamental Kalkulus

"Jadi, bedanya begini:

1. Visualisasi **irisan di bawah kurva** (dA) kita gunakan saat kita **TIDAK TAHU** sebuah fungsi luas ($A(x)$) dan kita ingin **MENEMUKAN SIFATNYA**.
2. Visualisasi **persegi yang ditambah** (df) kita gunakan saat kita **SUDAH TAHU** sebuah fungsi ($f(x)=x^2$) dan kita ingin **MENEMUKAN TURUNANNYA**.

Keduanya sama-sama pakai ide 'irisan tipis', tapi tujuannya berbeda."

Pertanyaan #3: Tentang Optimisasi (Gradien)

Bayangkan kamu sedang menjelaskan konsep Gradient Descent kepada adik kelasmu. Kamu menggunakan analogi "menuruni gunung dalam gelap". Adik kelasmu kemudian bertanya:

☰ Example scenario

"Oke kak, aku paham kalau Gradien itu kasih tau arah tanjakan tercepat, jadi kita ambil arah sebaliknya. Tapi, kalau kita udah deket banget sama dasar lembah, kan tanahnya jadi landai banget tuh. Berarti Gradiennya jadi kecil banget kan? Kalau gitu, langkah kita juga jadi kecil banget dong? Apa algoritmanya gak jadi **lambat banget** pas mau nyampe? Dan gimana dia tau kapan harus **berhenti**?"

Tugasmu:

Jawab pertanyaan adik kelasmu. Jelaskan apa yang terjadi pada **langkah-langkah Gradient Descent** saat mendekati titik minimum, dan jelaskan setidaknya **satu kriteria pemberhentian (stopping criterion)** yang logis untuk algoritma tersebut.

Jawaban:

Oke, setau aku itu tuh begini, kan memang ya, dalam gradient descent, kita tujuannya mencari dasar lembah paling terdalam. Nah apa yang terjadi apabila sudah dan sangat mendekati nilai maksimalnya dari dasar lembah itu? dia akan lama kelamaan berhenti, dia akan mencari dimana situasi nya akan semakin stabil, dan biasanya kita bisa menentukan suatu syarat berhenti apabila gradient descent tersebut tidak turun lagi dengan nilai misalnya. $0.5E10$ (berhenti apabila **perubahan nilai Loss** dari satu iterasi ke iterasi berikutnya lebih kecil dari ambang batas).

Tapi pernah terbenak di dalam pikiran tidak? gimana kalau misalnya ada lembah lagi yang ternyata lebih dalam lagi? gimana kalau ternyata kita kurang optimal jatuhnya?. Nah para peneliti memiliki berbagai cara, misalnya kek bola digelindingkannn sangat kencangg dari atas ke bawah, kan bisa saja apabila bola ternyata melewati puncak kedua (yang lebih rendah), dia akan melewati puncak tersebut dan jatuh dimana kemungkinan lembah tersebut lebih dalam lagi. Ada juga dengan coba cobaa dimana semoga aja jatuh di lembah paling dalam, dan lain sebagainya yang jauh dari konteks pemahaman kita.

Pertanyaan #4: Tentang Eigenvector & Eigenvalue

Kamu sedang mencoba menjelaskan Eigenvector kepada temanmu menggunakan analogi **transformasi sebagai "arus sungai"**. Kamu menjelaskan bahwa Eigenvector adalah "kayu yang nurut" yang hanya meluncur lurus mengikuti arus, tidak berbelok.

Temanmu mengerti, tapi kemudian bertanya:

Example Scenario

*"Oke, jadi Eigenvector itu 'kayu nurut'. Terus **Eigenvalue** itu apa? Cuma angka doang kan? Apa gunanya kita tahu 'kayu' itu jadi 3 kali lebih panjang atau cuma setengah kali panjangnya? Kenapa informasinya penting?"*

Tugasmu:

Jelaskan **kegunaan atau pentingnya nilai dari Eigenvalue**. Berikan setidaknya **satu contoh konkret** di mana **nilai** dari Eigenvalue (bukan hanya Eigenvector-nya) memberikan kita wawasan yang sangat berharga tentang apa yang dilakukan oleh sebuah transformasi. (Petunjuk: Pikirkan tentang contoh-contoh transformasi yang sudah kita bahas, seperti rotasi, scaling, dll.)

Oke, analogi “arus sungai”-nya sudah pas banget. Sekarang kita naik level dikit 😊

Kita mulai dari kalimat kunci ini dulu:

Kalau Eigenvector bilang arah mana yang “nurut”, maka Eigenvalue bilang seberapa keras arus sungainya di arah itu.

Eigenvalue itu “berapa kuat efek transformasinya”

Masih pakai analogi kayu di sungai 🌊 🪵

- **Eigenvector** → kayu yang *tidak belok* (arahnya tetap).
- **Eigenvalue** → apa yang terjadi pada kayu itu:
 - Memanjang?
 - Memendek?
 - Diam di tempat?
 - Bahkan kebalik arah?

Secara matematis:

$$A\mathbf{v} = \lambda\mathbf{v}$$

Artinya:

- Arah \mathbf{v} **tetap**
- Panjang \mathbf{v} **dikalikan λ (Eigenvalue)**

Kenapa Eigenvalue penting? (bukan cuma “angka doang”)

Karena **nilai Eigenvalue memberi informasi fisik / geometris yang sangat penting** tentang transformasi.

Mari kita lihat maknanya:

Nilai Eigenvalue (λ)	Makna di dunia nyata
$\lambda = 1$	Kayu tetap sama panjang (netral)
$\lambda > 1$	Kayu <i>ditarik</i> → diperbesar
$0 < \lambda < 1$	Kayu <i>ditekan</i> → diperkecil
$\lambda = 0$	Kayu “hancur” jadi titik
$\lambda < 0$	Kayu <i>dibalik arah</i>

Tanpa Eigenvalue, kita **tidak tahu seberapa ekstrem** efek transformasinya.

- **Eigenvector** → arah utama data
- **Eigenvalue** → *berapa besar variasi data di arah itu*

Kalau:

- Eigenvalue besar → arah itu **sangat penting**
- Eigenvalue kecil → arah itu **hampir tidak membawa informasi**

✳ Maknanya:

Eigenvalue memberi tahu **mana arah yang penting dan mana yang bisa dibuang**

Kalau harus dirangkum ke satu kalimat:

Eigenvector menjawab “ke mana”, Eigenvalue menjawab “seberapa besar dampaknya”.

Atau versi sungai:

Kayu boleh nurut, tapi **Eigenvalue bilang apakah dia cuma mengalir santai... atau diseret arus deras.**

Pertanyaan #5: Tentang Perubahan Basis (Menghubungkan Aljabar Linear & Geometri)

Kita sudah membahas dua cara untuk melakukan perubahan basis ke basis $B = \{b_1, b_2\}$:

1. **Metode Matriks Invers ($x_b = B^{-1} * x_e$):** Ini adalah metode umum yang selalu berhasil.
2. **Metode Dot Product (koordinat_i = $x_e \cdot b_i$):** Ini adalah "jalan pintas" yang sangat cepat.

Seorang temanmu melihat kedua metode ini dan bertanya:

☰ Example Scenario

"Aku bingung. Kenapa kita butuh metode matriks invers yang rumit itu kalau ternyata kita bisa pakai dot product yang jauh lebih gampang? Kenapa metode dot product tidak selalu bisa dipakai? Apa syarat spesialnya, dan kenapa syarat itu membuat dot product jadi berhasil?"

Tugasmu:

Jelaskan kepada temanmu **syarat wajib** agar "jalan pintas" dot product bisa digunakan. Lalu, jelaskan secara **intuitif atau geometris** kenapa syarat itu adalah kunci yang membuat metode proyeksi/dot product menjadi valid. (Petunjuk: Pikirkan tentang "bayangan" dan apa yang terjadi jika sumbu-sumbunya miring).

Jawaban:

Oke, bayangin lagi "Bahasa Alien" (b_1, b_2) yang sumbunya aneh itu. Metode Matriks Invers (B^{-1}) itu kayak punya aplikasi Google Translate yang canggih; dia bisa nerjemahin dari bahasa apa pun ke bahasa apa pun, gak peduli seberapa "aneh" tata bahasanya. Dia itu metode umum yang selalu berhasil.

Nah, metode Dot Product itu kayak "jalan pintas" atau "trik cepat". Trik ini cuma bisa berhasil kalau "bahasa" Alien-nya itu **"rapi"**.

Apa maksudnya "rapi"?

Syaratnya cuma satu: sumbu-sumbu acuannya (b_1 dan b_2) harus **saling tegak lurus (ortogonal)**.

Kenapa syarat itu penting banget?

Bayangin kamu berdiri di tengah ruangan dengan lantai kotak-kotak biasa (sumbu X dan Y tegak lurus). Kalau aku tanya, "Berapa jauh kamu ke arah Timur?", kamu bisa lihat "bayangan"-mu di sumbu X. Kalau aku tanya, "Berapa jauh kamu ke arah Utara?", kamu bisa lihat "bayangan"-mu di sumbu Y. **Bayangan di sumbu X sama sekali gak ngaruh ke bayangan di sumbu Y.** Mereka independen. **Proyeksi** (melihat bayangan) bekerja dengan sempurna di sini.

Sekarang, bayangin lantainya punya garis-garis yang **miring (tidak ortogonal)**. Ini adalah basis "berantakan"-nya si Alien.

- Sumbu b_1 miring ke kanan atas.
- Sumbu b_2 miring ke kiri atas.

Kalau sekarang aku tanya, "Berapa 'bayangan'-mu di sumbu b_1 ?", kamu bisa proyeksikan posisimu ke garis b_1 . **TAPI**, bayangan ini sekarang jadi "terkontaminasi". Karena sumbu b_2 juga miring, sebagian dari "jarak" di arah b_2 ikut menyumbang ke "bayangan" di arah b_1 . Mereka jadi saling tumpang tindih informasinya.

"Jalan pintas" Dot Product itu pada dasarnya adalah PROYEKSI. Metode ini hanya akan memberikan jawaban yang "jujur" dan akurat jika sumbu-sumbu acuannya **tidak saling mengganggu satu sama lain**, yaitu saat mereka **tegak lurus**. Kalau sumbunya miring, Dot Product akan memberikan jawaban yang salah karena "bayangan"-nya jadi lebih panjang atau pendek dari yang seharusnya. Makanya, untuk kasus "berantakan", kita harus pakai metode "Google Translate" (B^{-1}) yang lebih canggih.

Pertanyaan #6: Tentang Determinan (Menghubungkan Sifat & Konsekuensi)

Kamu sedang menjelaskan kepada adik kelasmu bahwa jika $\det(A) = 0$, maka matriks A tidak punya invers. Dia mengerti secara aljabar ("karena kita tidak bisa membagi dengan nol di rumusnya"). Tapi dia tidak mengerti **kenapa** secara konseptual.

Dia bertanya:

☰ Example Scenario

"Oke, jadi aku gak boleh bagi dengan nol. Tapi apa hubungannya itu dengan 'tidak punya invers'? Invers kan artinya 'aksi pembatal'. Kenapa kalau determinannya nol, aksinya jadi tidak bisa dibatalkan? Apa yang 'rusak' secara permanen?"

Tugasmu:

Jelaskan secara **geometris dan intuitif** kenapa sebuah transformasi dengan $\det(A) = 0$ tidak bisa "dibatalkan" (tidak punya invers). Gunakan analogi visual (misalnya, dari 2D ke 1D, atau 3D ke 2D) untuk menjelaskan informasi apa yang "hilang" secara permanen selama transformasi.

Jawaban:

Oke, $\det(A) = 0$ itu artinya "faktor pengali area/volume adalah nol". Secara visual, ini artinya transformasi A itu **"meremukkan" ruang**.

Bayangin kamu punya selembar kertas 2D yang penuh gambar (ruang input 2D). Transformasi A dengan $\det(A)=0$ itu kayak **mesin penghancur kertas** yang mengubah lembaran kertasmu menjadi **satu garis lurus tipis** (ruang output 1D).

Sekarang, "invers" (A^{-1}) itu kan artinya "aksi pembatal". Dia seharusnya bisa mengambil hasil akhir (garis lurus) dan **mengembalikannya** menjadi lembaran kertas bergambar seperti semula.

Kenapa itu mustahil?

Karena saat proses "meremukkan" itu terjadi, ada **informasi yang hilang secara permanen**.

Bayangin di kertas aslimu ada dua titik:

- Titik P: (3, 5)
- Titik Q: (3, 8)

Keduanya punya koordinat x yang sama, tapi y berbeda. Mereka ada di posisi vertikal yang berbeda.

Setelah dimasukkan ke mesin penghancur kertas yang meremukkan semuanya ke sumbu- X , kedua titik ini mungkin akan mendarat di **satu titik yang sama persis** di garis hasil, misalnya di (3, 0).

Sekarang, tugas si "mesin invers" adalah melihat titik (3, 0) di garis hasil dan mencoba menebak dari mana ia berasal.

- Apakah ia berasal dari (3, 5) ?
- Apakah ia berasal dari (3, 8) ?

- Atau dari $(3, -2)$?

Si mesin invers jadi bingung. Dia **tidak punya cukup informasi** untuk mengembalikannya ke satu posisi yang unik dan benar. Dia tidak bisa "menciptakan kembali" dimensi yang sudah hilang.

Karena transformasi A menyebabkan **kehilangan informasi yang tidak bisa dipulihkan**, maka tidak ada "aksi pembatal" (A^{-1}) yang bisa diciptakan. Itulah alasan geometris kenapa jika $\det(A)=0$, matriks A tidak punya invers.

Pertanyaan #7: Tentang Kalkulus (Menghubungkan Turunan & Limit)

Seorang teman melihat definisi formal turunan untuk pertama kalinya:

$$f'(x) = \lim_{h \rightarrow 0} \left[\frac{f(x+h) - f(x)}{h} \right]$$

Dia berkata:

Example Scenario

"Ini aneh banget. Katanya h mendekati nol, tapi tidak pernah benar-benar nol. Terus, kalau kita mau cari turunan x^2 , kenapa hasilnya bisa jadi $2x$ yang eksak? Bukannya seharusnya hasilnya cuma 'aproksimasi' atau 'hampir' $2x$? Bagaimana bisa proses 'mendekati' ini menghasilkan jawaban yang pasti dan tidak ada 'h'-nya sama sekali?"

Tugasmu:

Jelaskan kepada temanmu **keajaiban dari proses LIMIT** dalam perhitungan turunan. Gunakan contoh $f(x) = x^2$ untuk menunjukkan bagaimana, secara aljabar, suku h yang "mengganggu" pada akhirnya **lenyap secara sah**, sehingga kita mendapatkan jawaban yang eksak, bukan lagi aproksimasi.

Jawaban:

Kamu bener banget, ini emang kelihatan aneh. Rasanya kayak sulap, gimana bisa proses "mendekati" menghasilkan jawaban yang "pasti"?

Kuncinya ada di **proses aljabar** saat kita menjabarkan $f(x+h)$. Mari kita lihat lagi contoh $f(x) = x^2$.

Definisi turunan adalah:

$$\lim_{h \rightarrow 0} \left[\frac{(x+h)^2 - x^2}{h} \right]$$

Sekarang, ayo kita fokus ke bagian atas, $(x+h)^2 - x^2$. Ini adalah df , perubahan outputnya.

$$(x^2 + 2xh + h^2) - x^2 = 2xh + h^2$$

Jadi, rasio kita sekarang menjadi:

$$\lim_{h \rightarrow 0} \left[\frac{(2xh + h^2)}{h} \right]$$

Di sinilah "keajaiban" pertama terjadi. Lihat, setiap suku di bagian atas punya h . Jadi kita bisa **membagi semuanya dengan** h yang ada di bawah. Ini adalah langkah aljabar yang 100% sah, karena kita masih mengasumsikan h itu kecil, tapi **bukan nol**.

$$= \lim_{h \rightarrow 0} [2x + h]$$

Sekarang, kita sampai di langkah terakhir. Kita punya ekspresi yang jauh lebih sederhana: $2x + h$.

Baru **sekarang** kita ajukan pertanyaan "limit":

"Apa yang terjadi pada ekspresi $2x + h$ saat h menjadi semakin kecil dan semakin kecil, mendekati nol?"

- Bagian $2x$: Bagian ini sama sekali **tidak peduli** pada h . Mau h sekecil apa pun, $2x$ akan tetap $2x$.
- Bagian h : Bagian ini akan **menuju ke nol**.

Jadi, keseluruhan ekspresi $2x + h$ akan **menuju ke** $2x + 0$, yaitu $2x$.

Jadi keajaibannya bukan di proses limitnya, tapi di aljabarnya. Ternyata, untuk fungsi-fungsi "sopan" (seperti polinomial), setelah kita menjabarkan $f(x+h)$, kita akan selalu bisa **mencoret** h **di penyebut** dengan h yang muncul di setiap suku di pembilang.

Proses "limit" di akhir hanyalah "langkah pembersihan" untuk menghilangkan sisa-sisa suku h yang masih ada setelah pembagian. Itulah bagaimana proses "mendekati" bisa menghasilkan jawaban yang eksak.

Pertanyaan #8: Tentang Seluruh Gambaran (Menghubungkan Aljabar Linear & Kalkulus)

Ini adalah pertanyaan terakhir dan yang paling besar. Bayangkan kamu sedang menulis paragraf singkat untuk aplikasi beasiswa S2-mu. Kamu ingin menunjukkan pemahaman yang mendalam tentang hubungan antara Aljabar Linear dan Kalkulus dalam konteks Machine Learning.

Tugasmu:

Jelaskan dalam beberapa kalimat, **bagaimana konsep "Eigenvector" dari Aljabar Linear dan konsep "Gradient Descent" dari Kalkulus Multivariat sebenarnya saling berhubungan atau bahkan saling membutuhkan**. Mengapa pemahaman tentang "sumbu aksi" sebuah matriks (eigenvector) bisa relevan saat kita berbicara tentang "menuruni lembah" (gradient descent) di sebuah lanskap optimisasi? (Ini pertanyaan yang sulit dan terbuka, tidak ada satu jawaban benar. Yang aku cari adalah caramu menghubungkan kedua ide besar ini).

Jawaban:

Ini pertanyaan yang dalam banget. Di permukaan, keduanya kelihatan gak berhubungan sama sekali.

- **Eigenvector:** Tentang "sumbu aksi" dari sebuah **transformasi linear**.
- **Gradient Descent:** Tentang "menuruni lembah" dari sebuah **fungsi non-linear** yang rumit.

Tapi, jembatan penghubungnya adalah ini: **Perilaku Lokal**.

Ingat dari kalkulus, kalau kita "zoom in" sangat dekat ke sebuah kurva yang melengkung, kurva itu akan terlihat **seperti garis lurus**.

Hal yang sama berlaku untuk "lanskap" fungsi Loss kita di ML. Kalau kita "zoom in" sangat-sangat dekat ke satu titik di lanskap yang bergelombang itu, lanskapnya akan terlihat **hampir seperti mangkok parabola yang mulus**.

Nah, bentuk dari "mangkok parabola lokal" ini bisa dideskripsikan oleh sebuah **matriks**. Matriks ini disebut **Matriks Hessian** (turunan kedua dari fungsi Loss).

Di sinilah **Eigenvector** masuk.

- **Eigenvector dari Matriks Hessian** ini akan memberitahu kita "**sumbu utama**" dari mangkok lokal itu.
 - Eigenvector dengan eigenvalue terbesar akan menunjuk ke arah di mana mangkok itu **paling curam**.
 - Eigenvector dengan eigenvalue terkecil akan menunjuk ke arah di mana mangkok itu **paling landai**.
- **Eigenvalue dari Matriks Hessian** memberitahu kita "**tingkat kelengkungan**" di setiap arah sumbu itu.
 - Eigenvalue besar berarti mangkoknya sangat "sempit" dan curam di arah itu.
 - Eigenvalue kecil berarti mangkoknya sangat "lebar" dan landai.

Jadi, apa hubungannya dengan Gradient Descent?

Algoritma Gradient Descent yang "naif" hanya mengikuti arah $-\nabla f$ (turunan pertama). Dia seperti orang yang menuruni gunung dengan mata tertutup, hanya merasakan tanah di bawah kakinya.

Algoritma optimisasi yang **lebih canggih** (seperti *Newton's method*) menggunakan informasi dari **Matriks Hessian (turunan kedua)**. Mereka tidak hanya bertanya, "Ke mana arah turunan?", tapi juga bertanya, "Bagaimana bentuk lembahnya di sekitarku?"

Dengan mengetahui **Eigenvector dan Eigenvalue** dari Hessian, algoritma ini bisa mengambil langkah yang lebih cerdas.

- Dia bisa mengambil langkah yang lebih besar di arah yang landai (eigenvalue kecil).

- Dia bisa mengambil langkah yang lebih kecil dan hati-hati di arah yang curam (eigenvalue besar).

Jadi, **Eigenvector** (dari turunan kedua) memberikan **informasi tentang "geometri" dari lanskap** yang sedang coba dinavigasi oleh **Gradient Descent** (yang menggunakan turunan pertama). Keduanya saling melengkapi untuk menciptakan algoritma optimisasi yang lebih cepat dan lebih stabil.

Pertanyaan yang sangat bagus! "Konvergen" adalah istilah yang akan kamu dengar terus-menerus di dunia optimisasi dan Machine Learning.

Mari kita definisikan dengan analogi "menuruni gunung" kita.

Catatan - catatan tambahan

Apa Itu "Konvergen"?

Secara intuitif, sebuah proses disebut **KONVERGEN (CONVERGES)** jika ia **semakin lama semakin mendekati sebuah solusi akhir yang stabil**.

Bayangkan perjalanan pendaki kita (algoritma Gradient Descent):

- **Iterasi 1:** Posisi (10, 20)
- **Iterasi 2:** Posisi (8, 15)
- **Iterasi 3:** Posisi (6.5, 11)
- ...
- **Iterasi 100:** Posisi (0.1, 0.2)
- **Iterasi 101:** Posisi (0.08, 0.15)
- ...
- **Iterasi 1000:** Posisi (0.0001, 0.0002)
- **Iterasi 1001:** Posisi (0.00008, 0.00015)

Lihat polanya? Pendaki kita **semakin mendekat dan semakin mendekat** ke titik (0,0) . Langkah-langkahnya menjadi semakin kecil. Kita bisa bilang bahwa perjalanan pendaki ini **konvergen ke titik (0,0)** .

Apa Itu "Tidak Konvergen" (Divergen)?

Sebuah proses disebut **TIDAK KONVERGEN** atau **DIVERGEN (DIVERGES)** jika ia **tidak pernah "tenang" atau tidak pernah mendekati satu solusi akhir**.

Ada beberapa cara sebuah program bisa "tidak konvergen":

1. "Meledak" (Exploding)

- **Penyebab:** Biasanya karena **Learning Rate** terlalu besar.
- **Analogi:** Pendaki kita mengambil langkah yang terlalu besar. Dia melompati dasar lembah dan mendarat di lereng seberang yang lebih tinggi dari posisi awalnya.
- **Perilaku:**
 - Iterasi 1: `Loss = 100`
 - Iterasi 2: `Loss = 150` (Loh, kok jadi lebih buruk?)
 - Iterasi 3: `Loss = 500`
 - Iterasi 4: `Loss = 10000`
 - ...
 - Iterasi 10: `Loss = NaN` (Not a Number, karena angkanya sudah terlalu besar untuk dihitung komputer).
- Programnya "meledak" dan tidak akan pernah menemukan solusi.

2. "Memantul-mantul" (Oscillating)

- **Penyebab:** Learning rate juga terlalu besar, tapi tidak separah kasus pertama.
- **Analogi:** Pendaki melompati dasar lembah, tapi mendarat di ketinggian yang sama atau sedikit lebih rendah. Lalu dia melompat balik lagi, dan seterusnya.
- **Perilaku:**
 - Iterasi 1: `Loss = 100`
 - Iterasi 2: `Loss = 90`
 - Iterasi 3: `Loss = 95`
 - Iterasi 4: `Loss = 88`
 - Iterasi 5: `Loss = 92`
- Dia "membaik" secara umum, tapi nilainya terus memantul naik-turun dan tidak pernah benar-benar "tenang" di dasar lembah. Konvergensinya sangat lambat atau bahkan tidak terjadi.

3. "Bergetar" (Stuck)

- Ini terjadi di lanskap yang "kasar" atau "berkerikil".
- **Analogi:** Pendaki kita "menari-nari" di antara beberapa kerikil dan tidak pernah membuat kemajuan berarti menuju lembah utama.
- **Perilaku:** Nilai Loss-nya mungkin turun sedikit di awal, tapi kemudian berhenti berubah secara signifikan, meskipun masih sangat jauh dari solusi yang baik.

Kesimpulan:

Saat seorang engineer ML berkata:

- **"Modelnya konvergen"**: Ini adalah **kabar baik**. Artinya, proses training berjalan lancar, dan algoritma berhasil menemukan sebuah lembah (semoga yang dalam).
- **"Modelnya tidak konvergen"**: Ini adalah **kabar buruk**. Artinya, ada yang salah dengan proses training (kemungkinan besar *learning rate* atau arsitektur model), dan algoritma gagal menemukan solusi yang stabil.