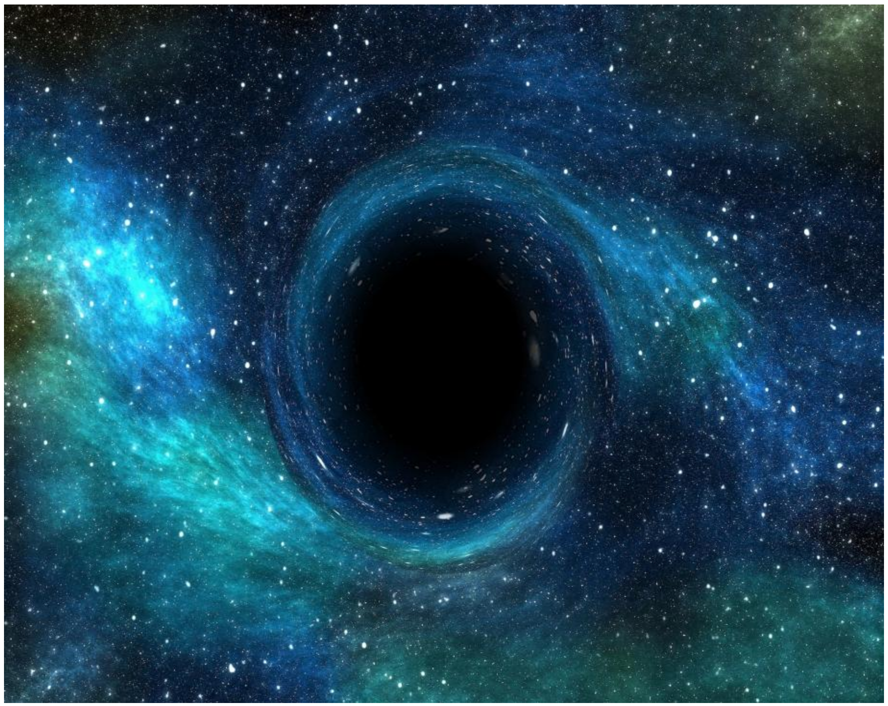
Linear Algebra

သတိ : စာကိုမကျက်ပါနှင့်။
နားလည်အောင်ဖတ်ပြီးစဉ်းစားပါ။

Space



What is Space?

ကျွန်တော်တို့ မေးခွန်းတွေနဲ့ စရအောင်ပါ။

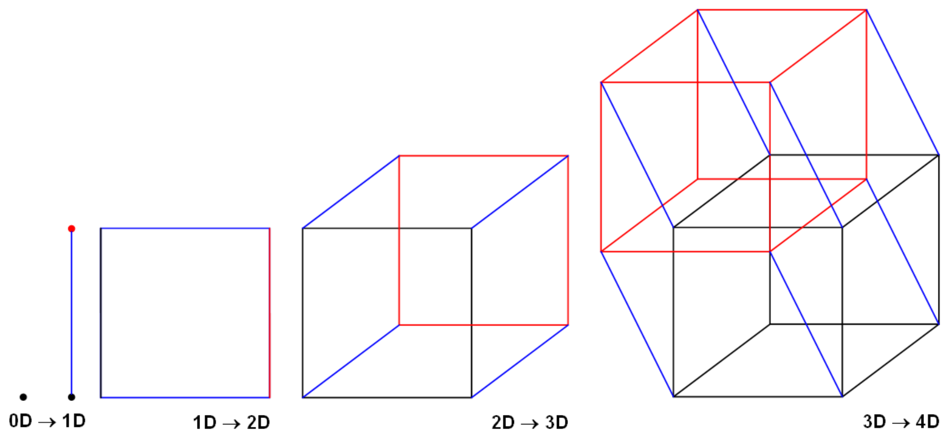
ကျွန်တော် Space လို့ ပြောတဲ့ ဟာကြီးထဲမှာ အသက်ရှင်နေကြပါသည်။ သို့သော် အဲဒါကြီးက ဘာကြီးလဲ။

ကျွန်တော်တို့သည် ကိုယ့်ပုံရိပ်ကို မှန်ထဲမှာ ပြန်မြင်ရသည်။ သို့သော် မှန်က အပြားကြီးဖြစ်သည်။ ကျွန်တော်တို့ကတော့ အပြားကြီး မဟုတ်။ ပုံရိပ် ဘယ်လိုပေါ်လာလဲ။

ပြီးတော့ ဓာတ်ပုံရိုက်သည်။ Camera က သေးသေးလေးဖြစ်သည်။ သို့သော် ကြီးမားသော တောင်ကြီးတွေ၊ သစ်ပင်တွေကို Camera ထဲမှာ ပုံပေါ်လာသည်။ ဘာကြောင့်လဲ။

ပြီးတော့ ကမ္ဘာလုံးကြီးကိုလည်း မြေပုံဆိုပြီး အပြားကြီးဆွဲလို့ရသည်။ ဘာကြောင့်လဲ။

Dimension



- အမှန်ပြောရရင် Space ဆိုတာ ဘာကြီးလဲ ကျွန်တော်ကိုယ်တိုင်လဲ မသိပါဘူး။ သို့သော် သင်္ချာသဘောတရား အရ ကျွန်တော်တို့ Dimension များကိုတော့ နားလည်နိုင်သည်။

- Dimension များသည် အတိုင်းအတာများသာ ဖြစ်သည်။ အချို့သော အတိုင်းအတာများကို တစ်ခုထဲ အနေဖြင့် ဖော်ပြလို့ ရသည်။ ဥပမာ၊ အသက်၊ အလေးချိန်။ သို့သော် အချို့သော အတိုင်းအတာများကို တစ်ခုထဲ ဖော်ပြလို့ မရ။ အတွဲလိုက် ဖော်ပြမှာသာ ရသည်။ ဥပမာ၊ ဆယ်တန်းအမှတ်စာရင်း။ ဆယ်တန်းအမှတ်စာရင်းကို ဖော်ပြမယ်ဆိုရင် မြန်မာစာအမှတ်၊ အင်္ဂလိပ်စာအမှတ် စသည်တို့ကို အတွဲလိုက် မပြောလို့ မရ။

- ဒီလို (Dimension) အတွဲလိုက် အတိုင်းအတာများကို ဖော်ပြတဲ့ သင်္ချာ သဘောတရား (Mathematical Concept) တစ်ခုကို Tensor ဟုခေါ်ပါသည်။

- Tensor နှင့် Set ဘာကွာလဲ ဆိုရင် Set က A Collection of Individuals (Entities) ဖြစ်ပြီး Tensor က A Collection of Components ဖြစ်သည်။

Tensor

- Tensor သည် N Dimensional Components များကို ဖော်ပြတဲ့ Mathematical Concept တစ်ခုကို ခေါ်ပါသည်။
- Computer နယ်ပယ်က လူတွေတော့ N-Dimensional Array တစ်ခုသည် Tensor တစ်ခုနဲ့တူပါသည်လို့ ပြောရင်တော့ နဲနဲသဘောပေါက်သွားကြမည် ဖြစ်သည်။
- သို့သော် Computation အရ Tensor ကို Array တစ်ခု အနေဖြင့် ယူဆနိုင်သော်လည်း အမှန်တော့ အဲဒါထက်ပိုပါသည်။
- ဓာတ်ပုံ (Picture) တစ်ခုကို 2 Dimensional Pixels Array ဖြင့် ဖော်ပြနိုင်သော်လည်း ဓာတ်ပုံတစ်ခုသည် Pixels Array တစ်ခု ထက်ပိုပါသည်။
- Tensor တစ်ခုသည်လည်း ထိုနည်းလည်းကောင်း ဖြစ်သည်။ ထို့ကြောင့် Tensor များရဲ့ ဆက်စပ်ပုံ၊ တွဲစပ်ပုံ၊ တွက်ချက်ပုံများ (Mathematical Operations on Tensors) ကို လေ့လာသော သင်္ချာပညာသည် Linear Algebra ဖြစ်ပါသည်။

Scalar

- Zero Dimensional Tensor ကို Scalar ဟု ခေါ်ပါသည်။
- Scalar များကို နားလည်ဖို့ လွယ်ကူပါသည်။
- အသက်၊ အလေးချိန် စသည်တို့ကို Scalar ဖြင့် ဖော်ပြလို့ရသည်။
- ဒါဆိုရင် အချိန်နာရီသည် Scalar တစ်ခု ဖြစ်နိုင်မလား။အချိန်နာရီမှာ နာရီ၊ မိနစ်၊ စက္ကန့်ဆိုပြီး 3 ခု ရှိသည့် အတွက်ကြောင့် Scalar ဖြစ်နိုင်ပါ့မလား။
- သို့သော်လည်း နာရီများကို စက္ကန့်ပြန်ဖွဲ့နိုင်သည့် အတွက်ကြောင့် နောက်ဆုံးမှာ တန်ဖိုး တစ်ခုထဲသာ ဖြစ်သည်။ ထို့ကြောင့် အချိန်နာရီသည် Scalar တစ်ခုသာ ဖြစ်သည်။
- Scalar များသည် Unit (Measurement)များ ကွဲချင်ကွဲမည်၊ သို့သော် အမြဲ Zero Dimension (တန်ဖိုးတစ်ခုတည်း) ဖြစ်သည်။

Vector

- One Dimensional Tensor ကို Vector ဟု ခေါ်ပါသည်။

- Vector သည် A List of Values <1,4,5,2,3,...> တစ်ခု ဖြစ်သည်။ Vector သည် Value တစ်ခုထက် ပိုပါဝင်သော Tensor တစ်ခု ဖြစ်သည်။

- အမှန်တော့ Vector တစ်ခုတွင်ပါဝင်သော (Components) တန်ဖိုးများသည် Orthogonal (ထောင့်မှန်) ဖြစ်ကြပါသည်။ တစ်နည်းအားဖြင့် တန်ဖိုးတစ်ခုကို အခြားတစ်ခုဖြင့် ဖော်ပြလို့ မရပါ။

- ဆယ်တန်း အမှတ်စာရင်းသည် Vector တစ်ခု ဖြစ်သည်။ မြန်မာစာ အမှတ်ကို အင်္ဂလိပ်စာအမှတ်ဖြင့် ဖော်ပြလို့ မရ။ အချိန်နာရီတွင်မူ နာရီကို မိနစ် သို့မဟုတ် စက္ကန့်ဖြင့် ဖော်ပြလို့ရသည်။

- Vector များသည် Tensor များ၏ အခြေခံဖြစ်သည်။ Higher Dimensional Tensor များကို Vector များအား တွဲစပ်ပြီး ဖော်ပြလို့ ရလို့ဖြစ်သည်။

Vector Space

- Vector များကို ဖော်ပြလို့ ရတဲ့ N Dimensional Space တစ်ခုကို Vector Space ဟုခေါ်ပါသည်။

- ကျွန်တော်တို့ ဆယ်တန်း အမှတ်စာရင်းကို စဉ်းစားကြည့်ရအောင်ပါ။ လွယ်ကူအောင် အမှတ်စာရင်းမှာ သင်္ချာအမှတ် (M)၊ မြန်မာစာအမှတ် (B)၊ အင်္ဂလိပ်စာအမှတ် (E)ပဲ ရှိသည်လို့ ဆိုပါတော့။

- **V** = <M, B,E> သည် Vector တစ်ခု ဖြစ်သည်။ ထို Vector ကို ကျွန်တော်တို့က Space တစ်ခုမှာ ဖော်ပြမည်ဆိုပါတော့၊ ထို Space သည် Vector Space တစ်ခု ဖြစ်ပါသည်။

- $\mathbf{V}_0 = <0, 0, 0>$, $\mathbf{V}_1 = < 50, 70, 60>$, $\mathbf{V}_2 = <30, 20, 40>$, …, $\mathbf{V}_n = <100,100,100>$

- $\mathbf{V}_0$ မှ $\mathbf{V}_n$ အထိသည် ရနိုင်သော အမှတ်စာရင်းအားလုံးကို ကိုယ်စားပြုသော Vector Space တစ်ခု ဖြစ်သည်။

- ဒီနေရာမှာ Space ကို သေသေချာချာ စဉ်းစားဖို့ လိုအပ်လာပါသည်။

Uniform Space

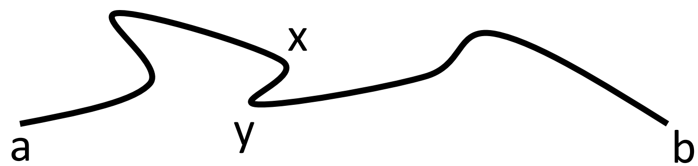
- a နှင့် b သည် မတူညီသော Number နှစ်ခု (b > a) ဖြစ်ပြီး a နှင့် b ကြားထဲတွင် ဖြစ်နိုင်သော Number များသည် Infinite ဖြစ်သည်။

- သိုသော် ကျွန်တော်တို့ ပြောနိုင်သည်က a နှင့် b ကြားထဲတွင် ဖြစ်နိုင်သော Number များသည် Uniform ဖြစ်သည်။ b > a ဖြစ်သည် အတွက်ကြောင့် b နှင့် နီးသော Number များသည် a နှင့် နီးသော Number များထက်ပိုကြီးပါသည်။ တစ်နည်းအားဖြင့် b နှင့် နီးလာလေလေ ပိုကြီးလာလေလေ ဖြစ်သည်။

- x နှင့် y သည် a နှင့် b ကြားရှိ မတူညီသော Number နှစ်ခု ဖြစ်ပြီး x သည် a နှင့် နီးပြီး y သည် b နှင့်နီးသည် ဆိုပါတော့။ ဒါဆိုရင် တကယ်လို့ y > x လို့ပြောနိုင်ရင်၊ x သည် တူညီစွာ y နှင့်နီးကပ်သွားရင် a နှင့် b ကြားရှိ Number များသည် Uniform ဖြစ်သည်ဟု ဆိုနိုင်ပါသည်။

- ထို့ကြောင့် a နှင့် b ကြားရှိ Space သည် Uniform ဖြစ်သည်။

Non Uniform Space



- a နှင့် b သည် မတူညီသော Number နှစ်ခု (b > a) ဖြစ်ပြီး a နှင့် b ကြားထဲတွင် ဖြစ်နိုင်သော Number များသည် Infinite ဖြစ်သည်။
- သိုသော် ကျွန်တော်တို့ ဒီတစ်ခါတော့ a နှင့် b ကြားထဲတွင် ဖြစ်နိုင်သော Number များသည် Uniform ဖြစ်သည်လို့ မပြောနိုင်တော့ပါ။ x နှင့် y သည် ဘယ်ဟာ a နှင့် ပိုနီးလဲ၊ b နှင့် ပိုနီးလဲ။ ဘယ်ဟာပိုကြီးလဲ။
- ဒီလို Space မျိုးကို Non-Uniform Space ဟု ခေါ်ပါသည်။
- ထို့ကြောင့် a နှင့် b ကြားရှိ Space သည် Non-Uniform ဖြစ်သည်။

Euclidean Space

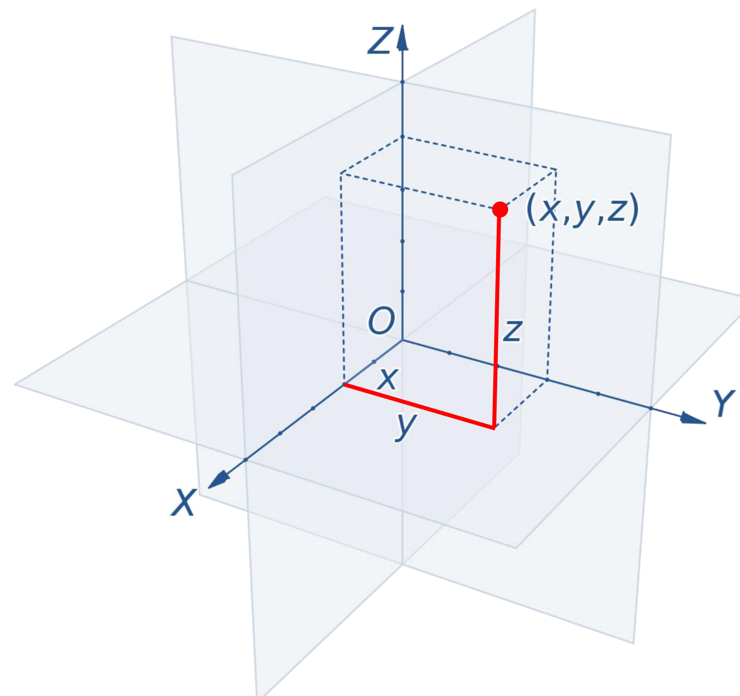
- Uniform ဖြစ်သော Space တစ်ခုကို Euclidean Space ဟုခေါ်ပါသည်။

- Euclidean Space လို့ ဘာလို့ ခေါ်လဲဆိုတော့ Euclidean Space မှာ Euclidean Geometry များ အလုပ်ဖြစ်လို့ပါ။ ဥပမာ၊ မျဉ်းပြိုင်နှစ်ကြောင်းသည် ဘယ်တော့မှ မထိပါ၊ တြိဂံ၏အတွင်းထောင့်များ ပေါင်းလဒ်သည် 180 ဒီဂရီ ဖြစ်သည် စသည်ဖြင့်ပေါ့။ (ဆယ်တန်းအထိ သင်ရသော Geometry သည် Euclidean Geometry ဖြစ်သည်။)

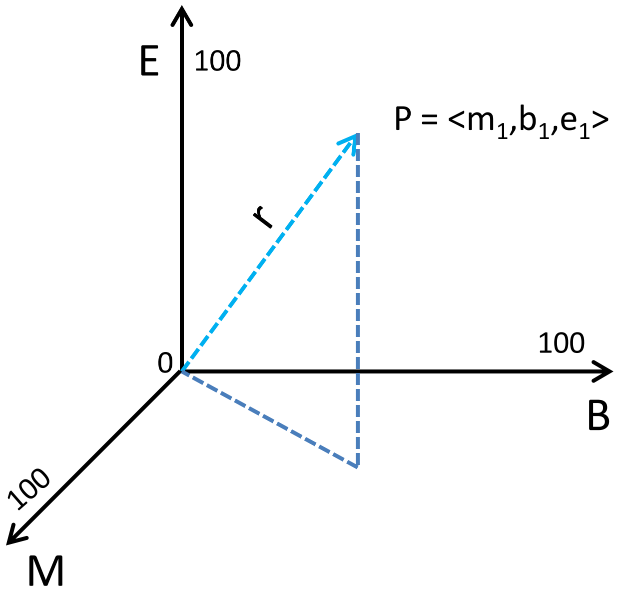
- သို့သော် Non-Euclidean Space မှာတော့ Euclidean Geometry များ အလုပ်မဖြစ်ပါ။ မယုံရင် ဘောလုံးတစ်လုံးပေါ်မှာ မထိနိုင်သော မျဉ်းပြိုင်များကို ဆွဲကြည့်ပါ။ ဘောလုံးမျက်နှာပြင်သည် Non-Euclidean Space တစ်ခု ဖြစ်သည်။

- အမှန်တော့ Vector Space သည် Linear Space အားလုံးကို ကိုယ်စားပြုပါသည်။ Euclidean Space ပဲ ဖြစ်ဖြစ်၊ Non-Euclidean Space ပဲဖြစ်ဖြစ်ပါ။

- သို့သော်လည်း ဘာမှမပြောရင်တော့ (Unless otherwise stated) Space ဆိုရင် Euclidean Space ကိုဆိုလိုတာ ဖြစ်သည်။



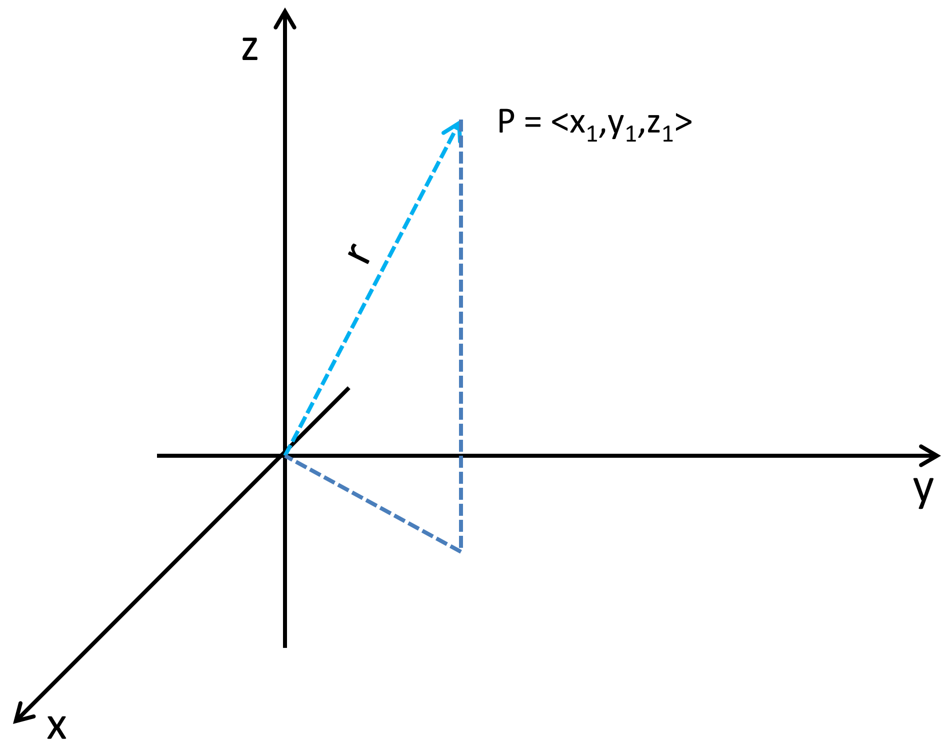
Example Vector Space



- အမှတ်စာရင်းရဲ့ Vector Space တစ်ခုကို Math, Burmese, English ဆိုပြီး Orthogonal Component 3 ခုဖြင့် ဖော်ပြနိုင်ပါသည်။

- ဒီလို Orthogonal Component များကို Orthogonal Basis ဟုခေါ်ပါသည်။ Vector Space တစ်ခု အတွင်းရှိ Vector အားလုံးကို Orthogonal Basis ဖြင့် ဖော်ပြလို့ ရပါသည်။

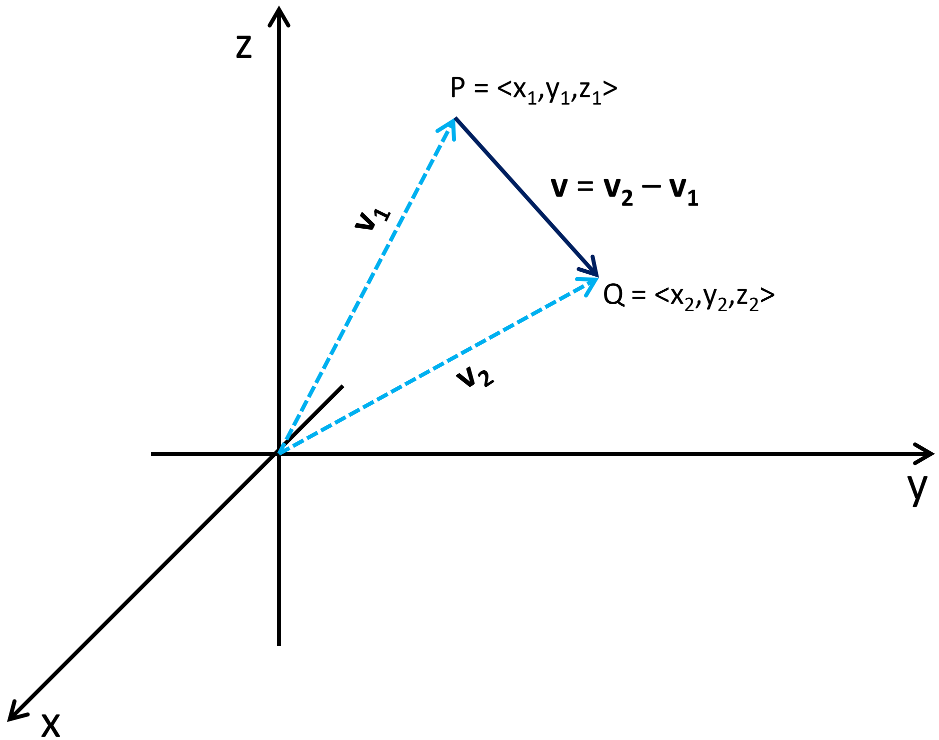
Position Vector



- Vector Space တစ်ခုမှာ Origin <0, 0, 0> မှ Point <x_1, y_1, z_1> တစ်ခုသို့ ဆွဲထားသော Directional Arrow တစ်ခုကို Position Vector **r** ဟုခေါ်သည်။

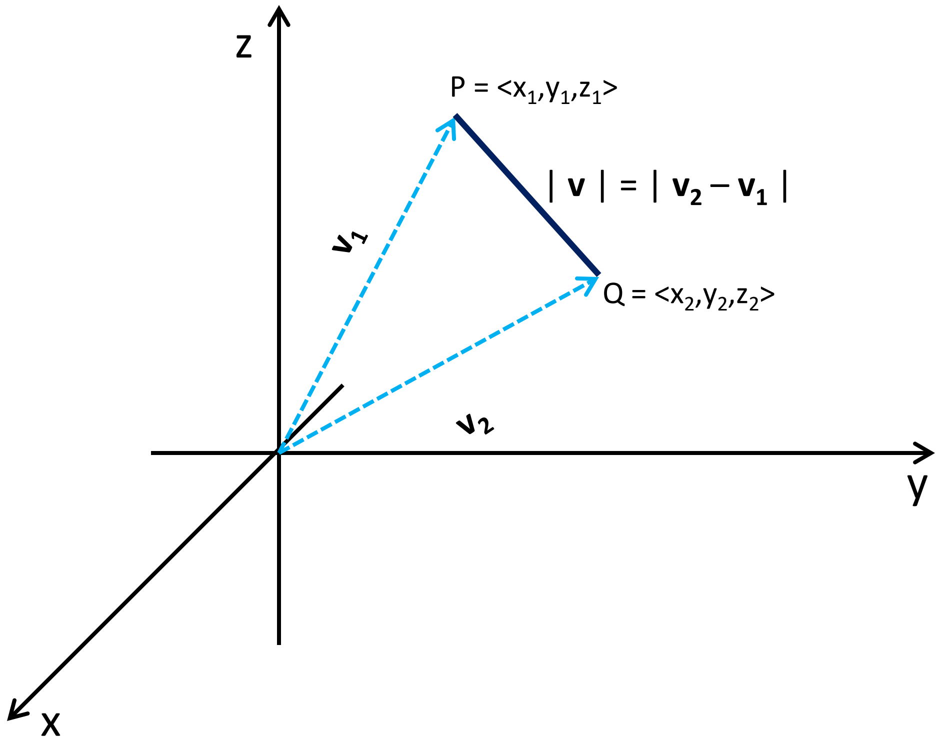
- Position Vector တစ်ခုသည် Origin <0, 0, 0> မှ ဘယ်လောက်ကွာဝေးပြီး ဘယ် Direction မှာ ရှိသည်ကို ကိုယ်စားပြုသည်။

Displacement Vector



- Displacement Vector သည် Position Vector နှစ်ခုကို ခြားနားခြင်းဖြင့် ရရှိသော Vector တစ်ခုဖြစ်သည်။

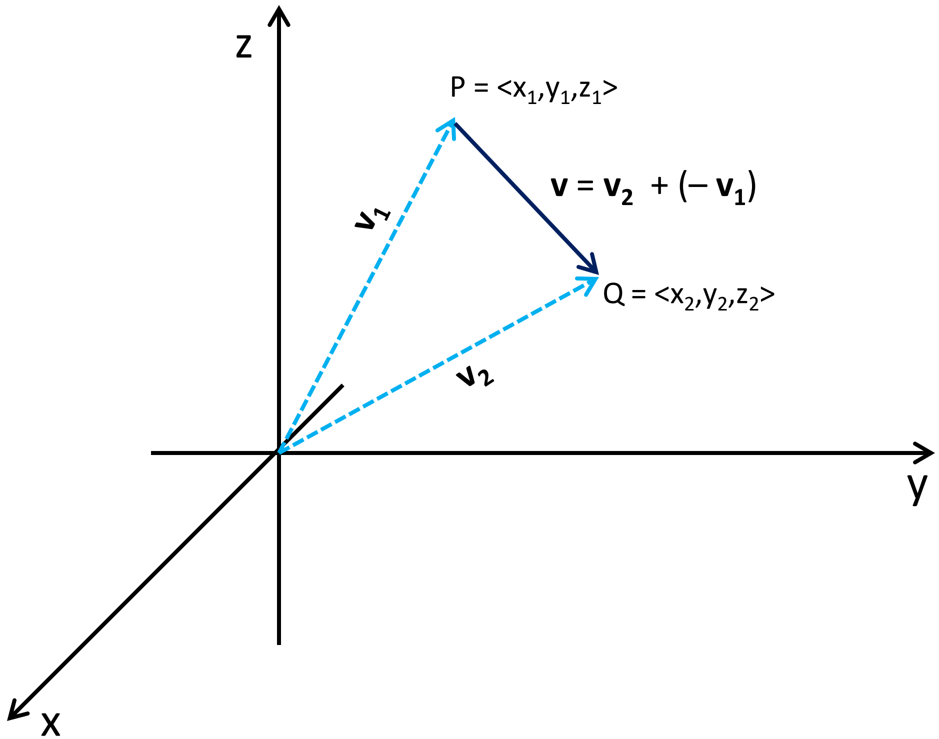
Magnitude of a Vector



- Magnitude of a Vector |**v**| သည် Distance between 2 Points in Vector Space ဖြစ်သည်။
- တစ်နည်းအားဖြင့် |**v**| သည် Vector **v** ၏ Length ကိုယ်စားပြုသည်။

|**v**| = Distance between **P** and **Q**

Linear Combination of Vectors



- Displacement Vector သည် Position Vector နှစ်ခုကို ခြားနားခြင်းဖြင့် ရရှိသော Vector တစ်ခုဖြစ်သည်လို့ ပြောခဲ့ပါသည်။

- ဒါဆိုရင် $\mathbf{v}$ ကို $\mathbf{v}_1$ နှင့် $\mathbf{v}_2$ တို့ဖြင့် ပြန်ဖော်ပြနိုင်ပါသည်။ $\mathbf{v} = \mathbf{v}_2 + (-\mathbf{v}_1)$ လို့ ဖော်ပြနိုင်ပါသည်။

- Vector တစ်ခုကို အခြား Vector များပေါင်းခြင်းဖြင့် ဖော်ပြခြင်းကို Linear Combination of Vectors ဟုခေါ်သည်။

Linear Combination of Vectors

$$\mathbf{v} = a_1\mathbf{v_1} + a_2\mathbf{v_2} + a_3\mathbf{v_3} + \dots + a_n\mathbf{v_n}$$

- Vector $\mathbf{v}$ တစ်ခုကို အခြား Vector များပေါင်းခြင်းဖြင့် ဖော်ပြခြင်းကို Linear Combination of Vectors ဟုခေါ်သည်။
- ထို့အတူ Vector $\mathbf{v}$ တစ်ခုကို အခြား Vector များပေါင်းခြင်းဖြင့် ဖော်ပြနိုင်လျှင် Vector $\mathbf{v}$ ကို Linearly Dependent of $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, …, $\mathbf{v}_n$ ဖြစ်သည်လို့ ပြောပါသည်။
- Orthogonal Basis Vectors များကို အခြား Vector များဖြင့် ဖော်ပြလို့ မရ။ ထို့ကြောင့် Orthogonal Basis Vectors များ Linearly Independent ဖြစ်သည်။

Linear Dependency of Vectors

$$\mathbf{v} = a_1\mathbf{v_1} + a_2\mathbf{v_2} + a_3\mathbf{v_3} + \ldots + a_n\mathbf{v_n}$$

- Vector **v** တစ်ခုကို အခြား Vector များပေါင်းခြင်းဖြင့် ဖော်ပြနိုင်လျှင် Vector **v** ကို Linearly Dependent of $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, …, $\mathbf{v}_n$ ဖြစ်သည်လို့ ပြောပါသည်။

- တကယ်လို့ အခြား Vector များပေါင်းလို့ **v** သည် Zero Vector တစ်ခု ဖြစ်သွားရင်

$$\mathbf{0} = a_1\mathbf{v_1} + a_2\mathbf{v_2} + a_3\mathbf{v_3} + \ldots + a_n\mathbf{v_n} \quad \text{if and only if } a_n = 0 \text{ where } n = 0,1,\ldots$$

- **v** ကို အခြား Vector များဖြင့် ဖော်ပြလို့ မရတော့ပါ။ ထို့ကြောင့် **v** သည် Linearly Independent ဖြစ်သည်။

- Orthogonal (ထောင့်မှန်) ဖြစ်သော Vector အားလုံးသည် Linearly Independent ဖြစ်သည်။

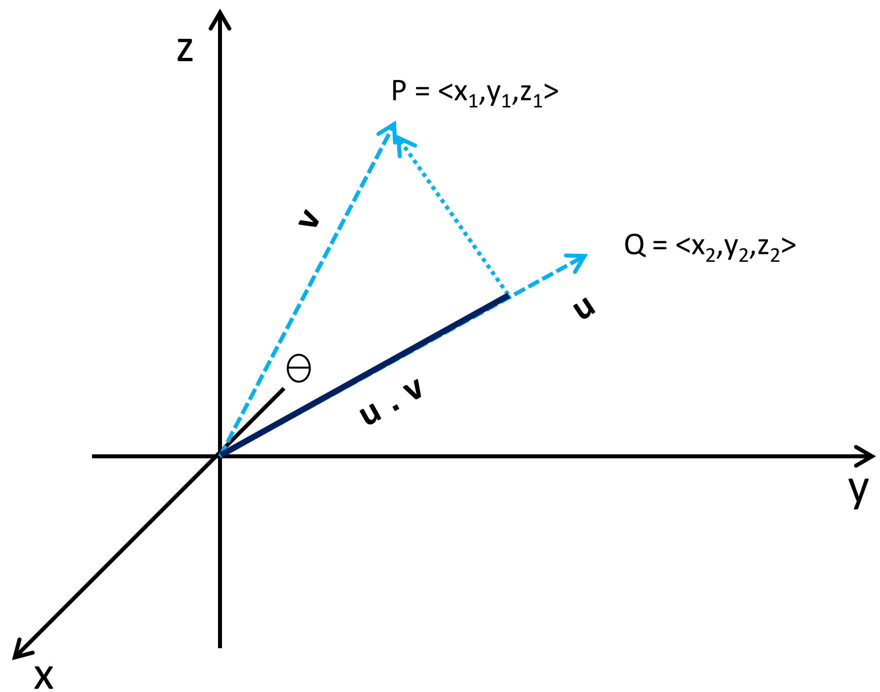
Dot Product of Vectors

$$\mathbf{u} \cdot \mathbf{v} = \mathbf{u}_1\mathbf{v}_1 + \mathbf{u}_2\mathbf{v}_2 + \mathbf{u}_3\mathbf{v}_3 + \dots + \mathbf{u}_n\mathbf{v}_n$$

- Vector **v** တစ်ခုကို အခြား Vector **u** နှင့် သက်ဆိုင်ရာ Component များကိုမြှောက်ပြီး ပေါင်းခြင်းဖြင့် Dot Product ကို ရရှိပါသည်။
- သင်္ချာအားဖြင့် ရိုးရှင်းသော်လည်း Dot Product သည် အဓိပ္ပါယ်အားဖြင့် လေးနက်ပါသည်။
- အမှန်တော့ Dot Product သည် Vector **v** နှင့် Vector **u** တို့၏ Linear Combination ကို ကိုယ်စားပြုပါသည်။
- ထို့ကြောင့် **u . v** = 0 ဆိုပါက Vector **v** နှင့် Vector **u** တို့သည် Linearly Independent ဖြစ်သည်၊ တစ်နည်း Orthogonal ဖြစ်သည်ကို ဆိုလိုပါသည်။
- တကယ်လို့ **u . v** သည် Zero မဟုတ်ပါက ဘယ်လို အဓိပ္ပါယ်ရှိမည်နည်း။

Dot Product of Vectors

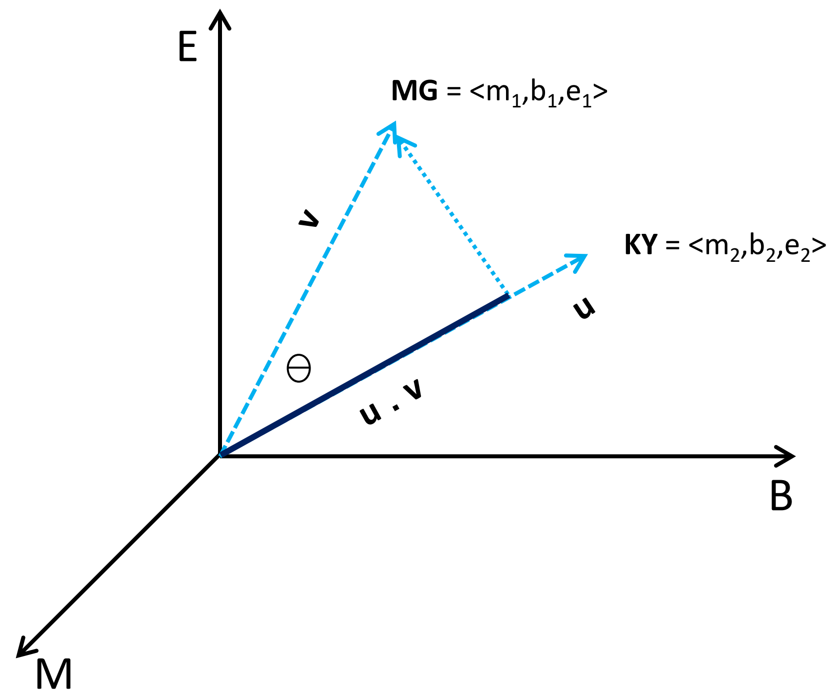
- တကယ်လို့ **u . v** သည် Zero မဟုတ်ပါက Dot Product သည် ထို Vector နှစ်ခု၏ Similarity (Projection) ကို ကိုယ်စားပြုပါသည်။
- တစ်နည်းအားဖြင့် Dot Product သည် Vector တစ်ခုကို အခြား Vector ဖြင့် ဘယ်လောက် ဖော်ပြနိုင်သည် ကို ကိုယ်စားပြုပါသည်။
- ပိုလွယ်အောင်ပြောရင်တော့ Dot Product သည် Component of **v** in the direction of **u** ဖြစ်သည်။
- Dot Product သည် Scalar တန်ဖိုး တစ်ခုဖြစ်သည်။



$$\mathbf{u} \cdot \mathbf{v} = |\mathbf{u}|\,|\mathbf{v}|\cos(\theta)$$

Applications of Dot Product

- ကျွန်တော်တို့ ဆယ်တန်း အမှတ်စာရင်းကို ပြန်ကြည့်ရအောင်ပါ။
- မောင်မောင်ရမှတ်က $\mathbf{MG} = <m_1, b_1, e_1>$ ဆိုပါတော့။
  မောင်မောင်၏ စုစုပေါင်းအမှတ်သည် $|\mathbf{MG}|$ ဖြစ်မည်။
- ကျော်ကျော်ရမှတ်က $\mathbf{KY} = <m_2, b_2, e_2>$ ဆိုပါတော့။
  ကျော်ကျော်၏ စုစုပေါင်းအမှတ်သည် $|\mathbf{KY}|$ ဖြစ်မည်။
- ဒါဆိုရင် မောင်မောင်နှင့် ကျော်ကျော် တို့
  ဘယ်လောက်အရည်အချင်းတူလဲလို့ ကြည့်ချင်ရင်
  စုစုပေါင်းအမှတ်ကို ကြည့်မည်။ သို့သော် မလုံလောက်သေးပါ။
  သူတို့ရဲ့ Similarity ပါကြည့်ရမည်။ မောင်မောင်က
  မြန်မာစာပိုတော်ပြီး ကျော်ကျော်က သင်္ချာပိုတော်တာ
  ဖြစ်နိုင်သည်။
- အခုခေတ်ကြီးမှာ Similarity ရှာတာသည် အင်မတန်
  အသုံးဝင်ပါသည်။
- Customer တစ်ယောက်က ပစ္စည်းတစ်ခုကို ကြိုက်သည်။
  ဒါဆိုရင် ဒါနှင့် အလားတူ ပစ္စည်းကို ထပ်ပြကြည့်မည်။
  Recommendation System များသည် Dot Product
  အပေါ် အခြေခံပါသည်။

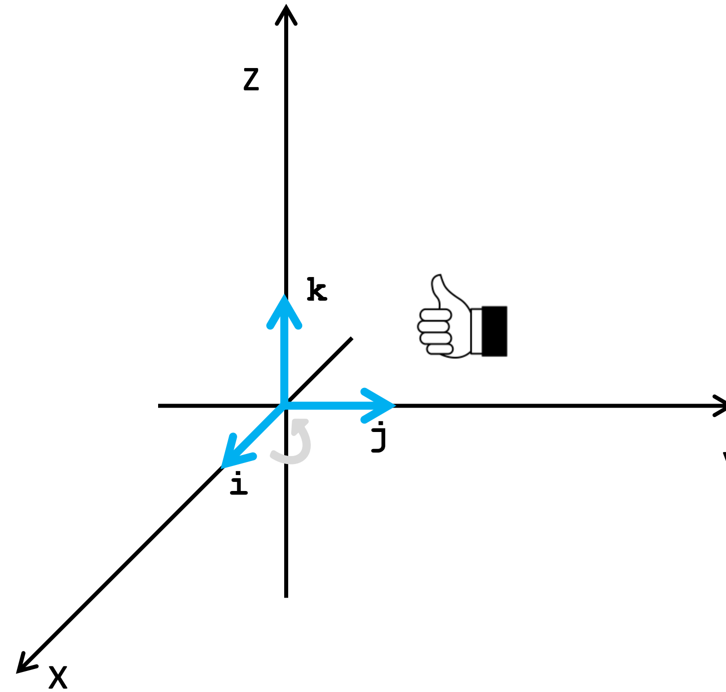


Cross Product of Vectors

- ကျွန်တော်တို့ 3 Dimensional Euclidian Space တစ်ခုကို စဉ်းစားကြည့်ရအောင်ပါ။

- **i, j, k** တို့သည် Unit Vector များဖြစ်ကြပြီး Unit Vector များ၏ Cross Product ကို Define လုပ်ပါသည်။ ဒါကို Right Thumb Rule လို့ ခေါ်ပါသည်။ လက်မထောင်ကြည့်ရင် လက်ချောင်းလေးများက Cross လုပ်မည် vector များကို ပြပြီး လက်မက result vector ကို ပြပါသည်။

- **v** သည် <a**i**, b**j**, c**k**> ဖြစ်ပြီး **u** သည် <d**i**, e**j**, f**k**> ဖြစ်သည် ဆိုပါတော့။

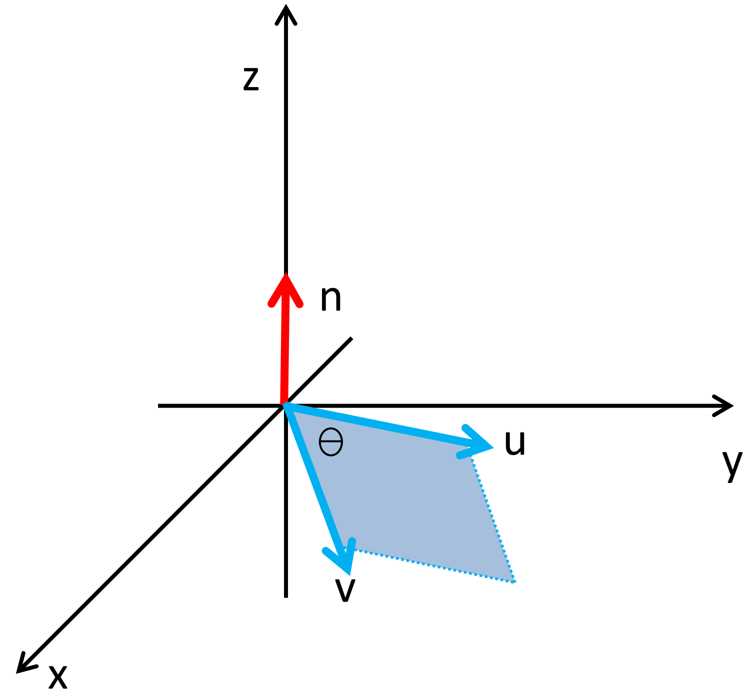



```
v × u = <ai, bj, ck> × <di, ej, fk>

      = (ad)(i × i) + (ae)(i × j) + … +
        (cf)(k × k)

      = (ae)k + (af)j − (bd)k + (bf)i −
        (cd)j − (ce)i
```

```
i × i =  0
j × j =  0
k × k =  0
i × j =  k
j × i = −k
j × k =  i
k × j = −i
i × k =  j
k × i = −j
```

Cross Product of Vectors

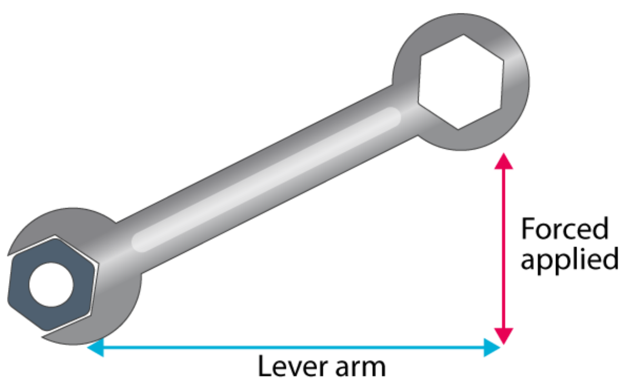
- |**u** × **v**| သည် Cross Product ကြောင့်ဖြစ်လာသော Parallelogram ၏ Area ကို ဖော်ပြပါသည်။
- ထို့ကြောင့် **u** နှင့် **v** သာ မျဉ်းတစ်ကြောင်းတည်း (Collinear) ဆိုလျှင် သို့မဟုတ် ပြိုင် (Parallel) နေလျှင် Cross Product သည် Zero ဖြစ်သည်။
- Cross Product သည် **u** နှင့် **v** တို့ Orthogonal ဖြစ်လျှင် တန်ဖိုးအများဆုံးရမည် ဖြစ်သည်။

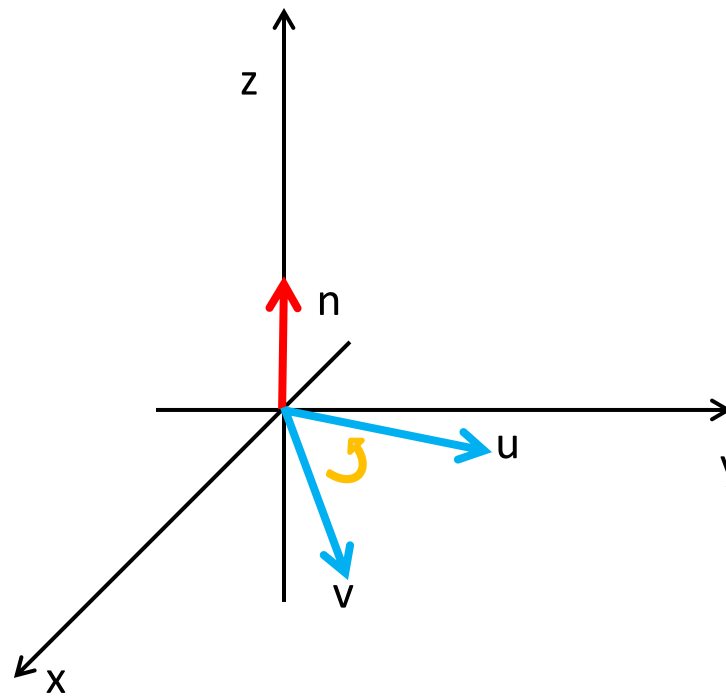


$$\mathbf{u} \times \mathbf{v} = |\mathbf{u}||\mathbf{v}|\sin(\Theta)\mathbf{n}$$

Application of Cross Product

- Cross Product ကိုအများအားဖြင့် Rotation များအတွက် အသုံးများပါသည်။
- ကျွန်တော်တို့ Curl ကို Advanced Calculus မှာ ပြောခဲ့ဖူးပါသည်။ Physics မှာ Torque ကို ပြောလျှင်လည်း Cross Product နှင့် ပြောရပါသည်။





Inner Product

- Inner Product သည် Dot Product နှင့် အတူတူပင် ဖြစ်ပြီး ပိုပြီး General ဖြစ်ပါသည်။
- Inner Product သည် Vector (One Dimensional Tensor) များသာမက Higher Dimensional Tensor များအတွက်ပါ ဖြစ်သည်။ ထို့ကြောင့် Dot Product ထက်ပိုပြီး General ဖြစ်သည်။
- သို့သော် Inner Product ၏ ပိုရှုပ်ထွေးသော အပိုင်းများကို နောက်ပိုင်းမှာ ထပ်ပြောပါမည်။
- Inner Product ၏ Concept သည် Linear Dependency ပေါ်မှာ အခြေခံထားတာ ဖြစ်သည့်အတွက် Linear Dependency or Independency သည် Linear Algebra မှာ အင်မတန် အရေးကြီးပါသည်။ Inner Product of a Tensor သည် အောက်ပါ အချက်များ ပေါ်မှာ အခြေခံပါသည်။

$$(\mathbf{u} + \mathbf{v}) \cdot \mathbf{w} = \mathbf{u} \cdot \mathbf{w} + \mathbf{v} \cdot \mathbf{w}$$

$$(a\mathbf{v} \cdot \mathbf{w}) = a(\mathbf{v} \cdot \mathbf{w})$$

$$(\mathbf{v} \cdot \mathbf{w}) = (\mathbf{w} \cdot \mathbf{v})$$

$$(\mathbf{v} \cdot \mathbf{v}) >= 0 \text{ if and only if } \mathbf{v} = 0$$

Outer Product

- Outer Product လို့ပြောတာသည် အမှန်တော့ Kronecker Product လို့ အသိများပါသည်။

$$\text{Outer Product of } \mathbf{v}, \mathbf{u} = \mathbf{v} \otimes \mathbf{u}$$

- Vector နှစ်ခုရဲ့ Outer Product သည် Matrix (Two Dimensional Tensor) တစ်ခု ရရှိမည် ဖြစ်သည်။

$$\mathbf{v} \otimes \mathbf{u} = \begin{pmatrix} v_1 u_1 & \cdots & v_1 u_n \\ \vdots & \ddots & \vdots \\ v_n u_1 & \cdots & v_n u_n \end{pmatrix}$$

Norm of a Vector

$$I_p = |\mathbf{v}|_p = (\mathbf{v}.\mathbf{v})^{\frac{1}{p}} = \sqrt[p]{\boldsymbol{v}.\boldsymbol{v}} \quad \text{where } p = 1,2,3,\ldots$$

- တကယ်လို့သာ P တန်ဖိုးသည် 2 ဖြစ်သည် ဆိုပါက I_2 ဖြစ်သွားပြီး ဒါကို Euclidean Norm (Distance) ဟု ခေါ်ပါသည်။
- Norm များသည် အမှန်တော့ Distance များကို ကိုယ်စားပြုခြင်း ဖြစ်ပြီး အများအားဖြင့် Error သို့မဟုတ် Mean (Average) များမှ Distance များကို ဖော်ပြရာတွင် သုံးလေ့ရှိပါသည်။
- Statistics မှ Standard Deviation သည် Euclidean Norm from Mean (Average) တစ်ခု ဖြစ်သည်။

Matrix

- Two Dimensional Tensor ကို Matrix ဟု ခေါ်ပါသည်။

- Matrix သည် ဇယားကွက် (Table) တစ်ခု ဖြစ်သည်။ Computer နယ်ပယ်က လူများအတွက်တော့ Matrix သည် 2 Dimensional Array တစ်ခုနှင့် အလားတူမည် ဖြစ်သည်။

- Matrix သည် 2 Dimensional Tensor သို့မဟုတ် Table ဖြစ်သည့်အတွက် Rows (တန်း:) နှင့် Columns (တိုင်) များပါဝင်ပါသည်။ n × m Dimensional ရှိသော Matrix A တစ်ခုကို ဒီလို ဖော်ပြကြပါသည်။

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{pmatrix}$$

Matrix

- တန်ဖိုးများကို တစ်ခုတည်း ဖော်ပြခြင်းထက် အတွဲလိုက် ဖော်ပြခြင်းအားဖြင့် တန်းဖိုးတစ်ခုတည်းက ဖော်မပြနိုင်တဲ့ ထူးခြားတဲ့ ဂုဏ်သတ္တိများကို ဖော်ပြနိုင်ကြောင်း Vector မှစပြီး ကျွန်တော်တို့ နည်းနည်း သဘောပေါက်လာပြီ ဖြစ်သည်။ ဒါသည် Tensor များ၏ ထူးခြားချက်ဖြစ်သည်။

- တန်ဖိုးများကို အတန်းပုံ အတွဲလိုက် (Vector ဖြင့်) ဖော်ပြရုံဖြင့် သူတို့ရဲ့ Space, Direction, Position, Distance, Similarity, Dependency, Rotation စတာတွေကိုပါ ကျွန်တော်တို့ သိရှိလာနိုင်ပါသည်။

- ထိုနည်းတူ တန်ဖိုးများကို ဇယားပုံ အတွဲလိုက် (Matrix ဖြင့်) ဖော်ပြခြင်းအားဖြင့် ဘယ်လို ထူးခြားတဲ့ ဂုဏ်သတ္တိများကို ဖော်ပြနိုင်မလဲ။

- ဒါကို ကျွန်တော်တို့ ဆက်ပြီး ဆွေးနွေးရအောင်ပါ။

Properties of Matrix

- Vector များ၏ Component များသည် တစ်ခုနှင့် တစ်ခု Orthogonal ဖြစ်ဖို့လိုအပ်သည်။

- Matrix များ၏ Component များသည် တစ်ခုနှင့် တစ်ခု Orthogonal ဖြစ်ဖို့တော့ မလိုအပ်ပါ။ ဒါပေမယ့် Orthogonal Component များသည် Matrix ၏ Rank ကို ကိုယ်စားပြုပါသည်။

- Matrix တစ်ခု၏ Rows သို့မဟုတ် Columns များကို Vector များအနေဖြင့် ယူဆနိုင်သည်။

$$\begin{pmatrix} 1 & 1 & 2 \\ 3 & 2 & 3 \\ 4 & 3 & 4 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 \\ 3 \\ 4 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \\ 4 \end{pmatrix}$$

- Matrix တစ်ခုမှာ Orthogonal ဖြစ်သော Row Vector သို့မဟုတ် Column Vector များ ဘယ်နှခုပါဝင်လဲသည် Matrix တစ်ခု၏ Rank ကို ကိုယ်စားပြုပါသည်။

- $n \times m$ Matrix တစ်ခုမှာ Orthogonal ဖြစ်သော Row Vector သို့မဟုတ် Column Vector 2 ခုရှိရင် Rank 2 Matrix ဖြစ်သည်။

Linear Function

- အရိုးရှင်းဆုံးသော Linear Function တစ်ခုကို ဒီလို ဖော်ပြနိုင်ပါသည်။

$$y = mx + b$$

- ပိုရှုပ်ထွေးသော Multivariable Linear Function တစ်ခုကိုတော့ ဒီလို ဖော်ပြနိုင်ပါသည်။

$$y = a_1x_1 + a_2x_2 + \dots + a_nx_n \text{ where } a_n = \text{parameters}$$

- ကျွန်တော်တို့သည် System of Linear Functions (Linear Function အများကြီးပါဝင်သော) တစ်ခုကို ဖြေရှင်းရမည် ဆိုပါတော့။

- ဥပမာ၊ ကျွန်တော်တို့ ငယ်ငယ်ကတွက်တဲ့ Algebra Problems (2 Unknowns, 2 Equations...)

$$\begin{aligned} 3x + 4y - 1 &= 6 \quad \dots (1) \\ 12x - 7y + 2 &= 10 \quad \dots (2) \end{aligned}$$

Linear Transformation

- ဒီလို System of Equations များကို ကျွန်တော်တို့ Matrix နှင့် Vector တို့ဖြင့် ဖော်ပြလို့ရပါသည်။

```
3x + 4y - 1  = 6   … (1)
12x - 7y + 2 = 10  … (2)
```

$$\begin{pmatrix} 3 & 4 \\ 12 & -7 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} -1 \\ 2 \end{pmatrix} = \begin{pmatrix} 6 \\ 10 \end{pmatrix}$$

- ဒါဆိုရင် ကျွန်တော်တို့သည Matrix နှင့် Vector တို့၏ Linear Function တစ်ခု ရလာမှာ ဖြစ်သည်။ Matrix နှင့် Vector တို့၏ Linear Function ကို Linear Transformation ဟုခေါ်ပါသည်။

$$\mathbf{y} = A\mathbf{x} + \mathbf{b}$$

- အမှန်တော့ Linear Transformation သည်ပင် Linear Algebra ကို အင်မတန် အသုံးများလာစေသော အချက်တစ်ခု ဖြစ်သည်။

- System of Equations နှင့် Linear Transformation များကို ကျွန်တော်တို့ နောက်ပိုင်း ပိုပြီးဆွေးနွေး ကြပါမည်။

Linear Combination of Vectors

- ဒီလို System of Equations များကို Linear Combination of Vectors အနေဖြင့်လည်း မြင်ကြည့်လို့ ရပါသည်။

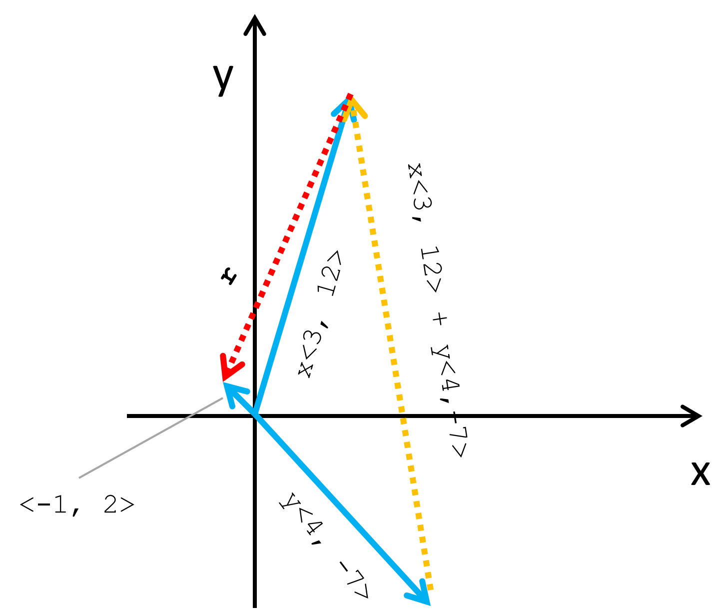
$$
\begin{aligned}
3x + 4y - 1 &= 6 \quad \ldots (1)\\
12x - 7y + 2 &= 10 \quad \ldots (2)
\end{aligned}
$$

$$x\begin{pmatrix}3\\12\end{pmatrix} + y\begin{pmatrix}4\\-7\end{pmatrix} + \begin{pmatrix}-1\\2\end{pmatrix} = \begin{pmatrix}6\\10\end{pmatrix}$$

တကယ်လို့ r သည် <6, 10> ဖြစ်ဖို့ x နှင့် y တန်ဖိုးသည် ဘယ်လောက်ဖြစ်ဖို့လိုမလဲ။

သို့သော်လဲ System of Equations များကို Linear Transformation အဖြစ်မြင်ကြည့်တာက ပိုအဆင်ပြေပါသည်။

အချို့ Application များမှာတော့ Linear Combination of Vectors အနေဖြင့် မြင်တာက ပိုအဆင်ပြေပါလိမ့်မည်။



Matrix Operations

- Matrix Operations များကို အကြမ်းအားဖြင့်

    1. Matrix တစ်ခု ၏ Element များကို Scalar တစ်ခုဖြင့် ပေါင်းခြင်း၊ နှုတ်ခြင်း၊ စားခြင်း၊ မြှောက်ခြင်း စသည့် Operations with Scalar။

    2. Matrix တစ်ခုကို အခြား Vector တစ်ခုနှင့် မြှောက်ခြင်း စသည့် Operations (Transformation) with Vectors။

    3. Matrix တစ်ခု ၏ Element များကို အခြား Matrix တစ်ခု ၏ Element များနှင့် ပေါင်းခြင်း၊ နှုတ်ခြင်း၊ စားခြင်း၊ မြှောက်ခြင်း စသည့် Operations on Elements။

    4. Matrix တစ်ခုကို အခြား Matrix တစ်ခု နှင့် မြှောက်ခြင်း စသည့် Operations (Matrix Multiplication) with Matrix ။

    5. Matrix တစ်ခုကို Inverse လုပ်ခြင်း၊ Transpose လုပ်ခြင်း စသည့် Operations with itself ။

Matrix Operations with Vectors

- Linear Transformation တစ်ခုကို ကျွန်တော်တို့ ဒီလို ဖော်ပြလို့ရကြောင်းပြောခဲ့ပါသည်။

$$\mathbf{y} = A\mathbf{x} + \mathbf{b}$$

- ကျွန်တော်တို့ $\mathbf{b}$ ကို ဖယ်ထားလိုက်ရင်

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

- $n \times n$ Matrix A နှင့် n Dimensional Vector $\mathbf{x}$ တို့ မြှောက်လိုက်လျှင် n Dimensional Vector $\mathbf{y}$ ကို ရရှိမည် ဖြစ်သည်။

$$[n \times n]\,[n] = [n]$$

- တကယ်လို့ Matrix A သည် $n \times n$ မဟုတ်ဘဲ $m \times n$ ဆိုပါတော့ ဒါဆိုရင် ရလာသော Vector သည် m Dimensional Vector ဖြစ်သွားမည်။

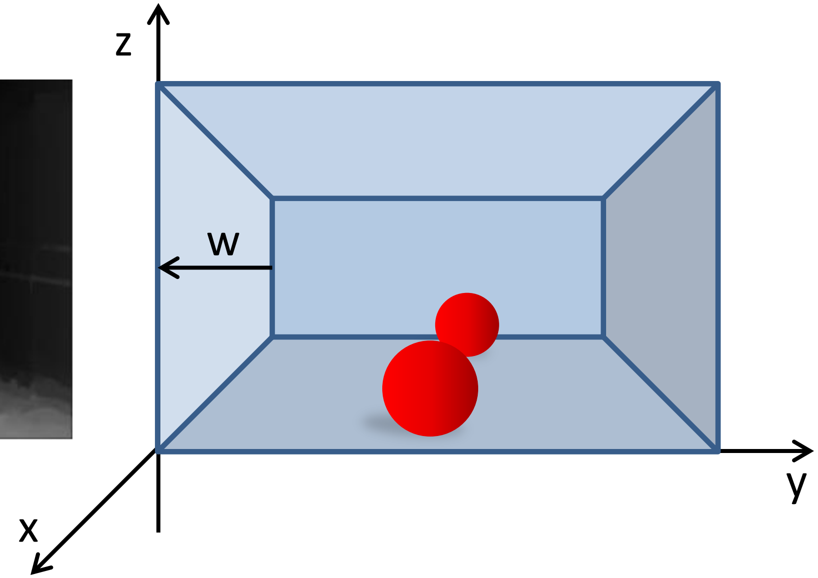
$$[m \times n]\,[n] = [m]$$

- ဒီအချက်သည် အင်မတန် အရေးကြီးပါသည်။ ဒါကို သုံးပြီး n Dimensional Vector $\mathbf{x}$ ကို m Dimensional Vector $\mathbf{y}$ သို့ပြောင်းပေးနိုင်ပါသည်။

- ကျွန်တော် အစမှာ ပြောခဲ့သလို မှန်ထဲမှာ အရိပ်ပေါ်ခြင်း၊ ကမ္ဘာလုံးကြီးကို မြေပုံဆွဲနိုင်ခြင်းသည် n Dimensional Vector $\mathbf{x}$ ကို m Dimensional Vector $\mathbf{y}$ သို့ပြောင်းပေးနိုင်လို့ ဖြစ်သည်။ တစ်နည်းအားဖြင့် n Dimensional Space တစ်ခုမှ m Dimensional Space သို့ပြောင်းပေးလိုက်ခြင်း ဖြစ်သည်။ လက်တွေ့မှာ Space များကို Dimension တစ်ခုမှ တစ်ခုသို့ ကူးပြောင်းနိုင်ခြင်း (Transformation)သည် အင်မတန် အသုံးဝင်ပါသည်။

Homogenous Coordinates

(a) Real-image (b) Depth-map



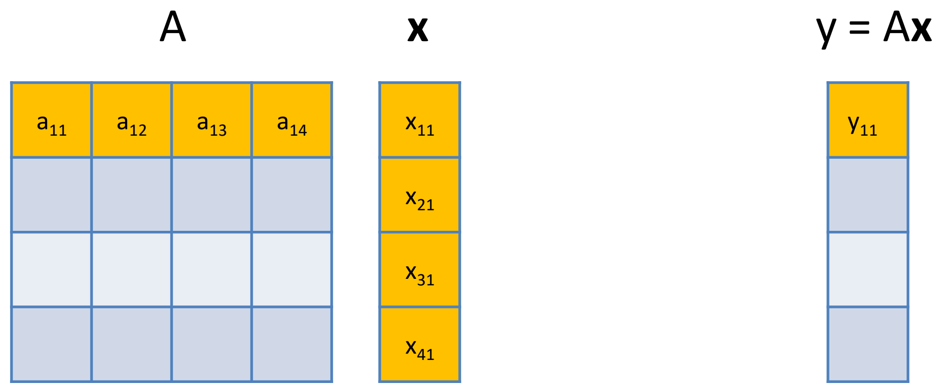
- ကျွန်တော်တို့သည် 3 Dimensional Space ထဲမှာ နေရသော်လည်း (နေရသည်ဟု ထင်ရသော်လည်း) အမှန်တော့ ကျွန်တော်တို့ ပတ်ဝန်းကျင်လောကကြီးကို (မျက်စိမှ) မြင်ရသည်က 2 Dimension ဖြစ်ပါသည်။

- 2 Dimension ဖြစ်သော်လည်း 3^{rd} Dimension ကို Perspective Vision ဆိုပြီး Depth Information တစ်ခု အနေဖြင့် မြင်ရပါသည်။ ဝေးသော Object များသည် သေးသွားမည်၊ မျဉ်းပြိုင်များသည် မိုးကုတ်စက်ဝိုင်းမှာ ဆုံသွားမည် စသည်တို့သည် Perspective Vision ကြောင့် ဖြစ်ပါသည်။

- ဒါကို Homogenous Coordinates (x, y, z and w) ဖြင့် ဖော်ပြကြပါသည်။ w သည် Depth ကို ကိုယ်စားပြုတာ ဖြစ်သည်။

- ဒါကို ဘယ်မှာ သုံးလဲဆိုတော့ Computer Graphics (Computer Games and Simulation), Machine Vision နှင့် AR (Augmented Reality), VR (Virtual Reality) လုပ်သူများအတွက် မရှိမဖြစ် သုံးပါသည်။

Matrix-Vector Multiplication



- Resultant Vector **y** ၏ Component တစ်ခုခြင်းစီသည် Matrix A ၏ Row Vector များနှင့် Vector **x** တို့၏ Dot Product ဖြစ်သည်။

if $A = [m \times n]$ Matrix and $\mathbf{x} = [n \times 1]$ Vector , then $\mathbf{y} = [m \times 1]$ Vector

$y_i = A[\mathbf{r}_i] \cdot \mathbf{x}$ where $A[\mathbf{r}_i]$ = Row Vector of A

Matrix Multiplication

- Function တစ်ခုရဲ့ Function (Composite Function) သည်

$$f(g(x)) = (f \cdot g)(x)$$
$$f(g(x)) \neq g(f(x))$$

- ဒါဆိုရင် Matrix တစ်ခုကို Linear Transformation (Function) တစ်ခုလို့ယူဆရင်၊ Composite Linear Transformation သည် ဘာဖြစ်မလဲ။

- Composite Linear Transformation သည် Matrix Multiplication ပါပဲ။

- Matrix နှစ်ခု မြှောက်ခြင်းသည် Composite Function နှင့် အလားတူပါသည်။

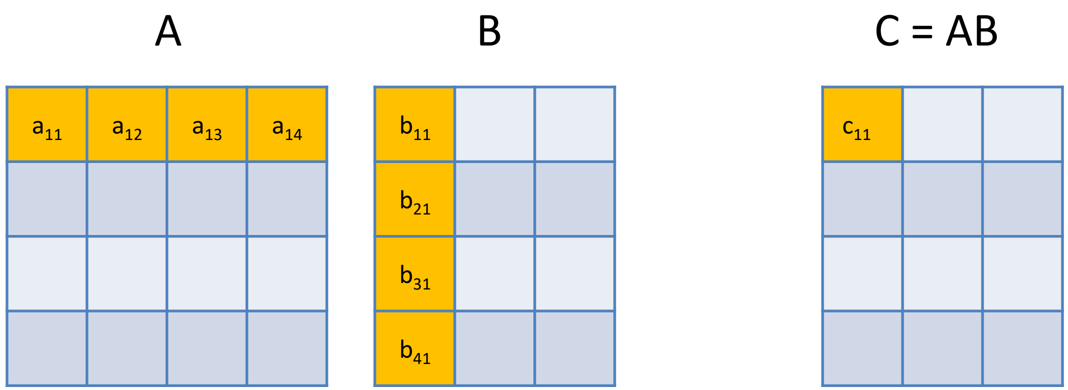
$$(\text{Composite Transformation of A and B})\mathbf{x} = AB\mathbf{x}$$
$$AB\mathbf{x} \neq BA\mathbf{x}$$

$$[m \times n]\ [n \times m] = [m \times m]$$

$$[n \times n]\ [n \times m] = [n \times m]$$

$$[m \times m]\ [n \times n] = \text{undefined}$$

Matrix-Matrix Multiplication

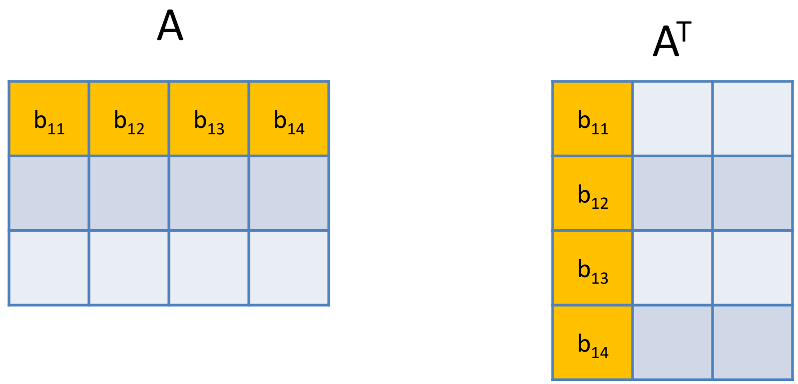


- Resultant Matrix C ၏ Element တစ်ခုခြင်းစီသည် Matrix A ၏ Row Vector များနှင့် Matrix B ၏ Column Vector များတို့၏ Dot Product ဖြစ်သည်။

if A = [m × n] Matrix and B = [n × r] Matrix, then C = [m × r] Matrix

$C_{ij} = A[\mathbf{r}_i] \,.\, B[\mathbf{c}_j]$ where $A[\mathbf{r}_i]$ = Row Vector of A and $B[\mathbf{c}_j]$ = Column Vector of B

Matrix Transposition



- Transpose of a Matrix ဆိုသည်မှာ Matrix တစ်ခု၏ Rows နှင့် Columns များကို နေရာပြောင်းလိုက်ခြင်း ဖြစ်သည်။

$$[A]_{ij} = [A^T]_{ji}$$

Inverse of a Matrix

- $f(x)$ သည် function တစ်ခုသာဆိုရင် $f^{-1}(x)$ သည် $f(x)$၏ Inverse Function တစ်ခု ဖြစ်သည်။
- ထို့အတူ Matrix A ကို Linear Function (Transformation) လို့ ယူဆလိုက်ရင် A^{-1} သည် Matrix A ၏ Inverse Matrix ဟုခေါ်သည်။
- သို့သော် Matrix တိုင်းမှာ Inverse မရှိပါ။
- Matrix တစ်ခုမှာ Inverse ရှိဖို့ Square Matrix $[n \times n]$ ဖြစ်ရပါမည်။ ထို့အတူ Matrix တစ်ခု၏ Rank (Orthogonal Row/Column Vectors) များသည်လည်း n ဖြစ်ရပါမည်။
- Square Matrix မဟုတ်သော Matrix $[n \times m]$ များတွင် Inverse မရှိပါ။ သို့သော် Pseudo Inverse ဆိုပြီးတော့ ရှာကြရပါသည်။ ဇယားကွက် (Matrix) များသည် အမြဲ Square မဖြစ်နိုင်သည့် အတွက် Pseudo Inverse များလက်တွေ့မှာ အသုံးဝင်ပါသည်။ Pseudo Inverse များကို နောက်ပိုင်းမှာ ထပ်ပြောပါမည်။

$$AA^{-1} = A^{-1}A = I$$

Singular Matrix

- Square Matrix [n × n] တစ်ခု၏ Dimension များသည် ၎င်း၏ Rank နှင့် မတူညီပါက ထို Matrix တွင် Inverse မရှိပါ။

- Rank နှင့် Dimension မတူညီသော Square Matrix များကို Singular Matrix ဟုခေါ်ပါသည်။

- တစ်နည်းအားဖြင့် Square Matrix [n × n] တစ်ခု၏ Row Vectors သို့မဟုတ် Column Vectors များအားလုံးသည် Linearly Independent (Orthogonal) မဖြစ်ပါက ထို Square Matrix ကို Singular Matrix ဟုခေါ်ပါသည်။

Determinant

- Matrix တစ်ခု၏ Inverse ကို ကျွန်တော်တို့ ပြောခဲ့ပါသည်။ ဒါဆိုရင် ထို Inverse ကို ဘယ်လို ရှာမလဲ။
- ဒါဆိုရင် ကျွန်တော်တို့ Determinant of a Matrix ကို ပြောရပါမည်။
- အမှန်တော့ Determinant သည် Inverse ရှာဖို့တင်မကပါ။ အခြားနေရာများစွာ မှာပါ အသုံးဝင်ပါသည်။

$$\det(A) = |A| = \begin{vmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{vmatrix}$$

Determinant By Laplace Expansion

- Laplace Expansion အရ

$$\det(A) = |A| = \sum_{j=1}^{n} a_{ij}C_{ij}$$

- C_{ij} ကို Cofactor ဟုခေါ်သည်။ Cofactor ကို Determinant of Minor Matrix (M) များမှ တွက်ယူပါသည်။ Minor Matrix များသည် a_{ij} ၏ Row နှင့် Column မပါတဲ့ Sub Matrix များဖြစ်သည်။

- တစ်နည်းအားဖြင့် det(A) သည် $\det(M_{ij})$ များ၏ Linear Combination ဖြစ်သည်။ $\det(M_{ij})$ ကိုလည်း သူရဲ့ Sub Matrix များမှ ပြန်တွက်ယူမည်။

- Laplace Expansion အရ Determinant ကို Recursively ပြန်တွက်လို့ ရပါသည်။ သို့သော် တွက်လို့ ပိုလွယ်တဲ့ အခြားနည်းလမ်း ရှိပါသေးသည်။

$$C_{ij} = (-1)^{i+j}|M_{ij}| \quad , \text{where } |M_{ij}| = \det(M_{ij})$$

Determinant By Leibniz Formula

- Leibniz Formula အရ

$$\det(A) = |A| = \sum\left(sgn(\sigma)\prod_{i=1}^{n} a_{i},\sigma_{i}\right)$$

- Determinant ကို Permutation (P) နှင့် Signature Function (sgn) တို့မှ ပြန်တွက်ယူပါသည်။

Signature Function (sgn) သည် Inversion of a Pair of Number (i, k) အပေါ် အခြေခံပါသည်။

k သည် i ထက်ကြီးသော်လည်း i က k အရှေ့မှာ ရှိပါက Inversion ဖြစ်သည်လို့ ဆိုလိုသည်။

$$sgn(\sigma) = (-1)^{n} \quad , \text{ where } n = \text{Number of inversion}$$

P	$sgn(\sigma)$	$\prod_{i=1}^{n} a_{i},\sigma_{i}$
1,2,3	+1	$a_{11}a_{22}a_{33}$
1,3,2	-1	$a_{11}a_{23}a_{32}$
2,1,3	-1	$a_{12}a_{21}a_{33}$
2,3,1	+1	$a_{12}a_{23}a_{31}$
3,1,2	+1	$a_{13}a_{21}a_{32}$
3,2,1	-1	$a_{13}a_{22}a_{31}$

Interpretation of Determinant

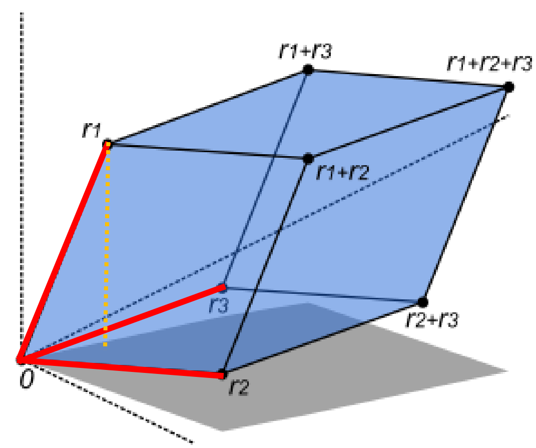
- အခု Determinant ကို ရှာဖို့ နည်း 2 နည်းသိသွားပါပြီ။ ဒါဆိုရင် Determinant ဆိုတာ ဘာကြီးလဲလို့ စဉ်းစားကြည့်ရအောင်ပါ။

- Matrix A ရဲ့ Row Vector များကို မြင်ကြည့်ပါ။ ထို Row Vector များက Bound လုပ်ထားတဲ့ Region ရဲ့ Volume သည် Determinant ဖြစ်သည်။

- တကယ်လို့ ထို Row Vector များက Linearly Independent ဖြစ်မနေဘူး ဆိုရင် ($\mathbf{r}_1$ သည် $\mathbf{r}_2$ နှင့် $\mathbf{r}_3$ တို့ပေါင်းလဒ်ဖြင့် ဖော်ပြလို့ ရမည်ဆိုပါက $\mathbf{r}_1$ သည် $\mathbf{r}_2$ နှင့် $\mathbf{r}_3$ တို့၏ Plane ပေါ်ရောက်သွားပြီး $\mathbf{r}_1$ ၏ Orthogonal Component ပျောက်သွားမည်) ထို Row Vector များက Bound လုပ်ထားတဲ့ Region ရဲ့ Volume သည် Degenerate (ပုံပျက်သွားမည်) ဖြစ်သည်။ ထိုအခါ Determinant မရှိတော့ဘဲ Zero ဖြစ်သွားမည်။

- ထို့ကြောင့် Singular Matrix များ၏ Determinant သည် Zero ဖြစ်နေတာ ဖြစ်သည်။

$$\det(A) = |A| = \begin{vmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{vmatrix}$$



Inverse of a Matrix

- Inverse of a Matrix ကို Laplace Expansion မှ

$$A^{-1} = \frac{adj(A)}{\det(A)}$$

- C သည် Cofactor Matrix ဖြစ်သည်

$$C = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{m1} & \cdots & c_{mn} \end{pmatrix}$$

- Adjoint of a Matrix, adj (A)သည် Cofactor Matrix ၏ Transpose ဖြစ်သည်။

$$adj(A) = C^T$$

More Linear Algebra

- ကျွန်တော်တို့ အခုထိပြောခဲ့တာတွေသည် Linear Algebra ၏ အခြေခံများ ဖြစ်သည်။

- Linear Algebra ဆိုသည့်အတိုင်း Linear Dependency ကို နေရာတကာ နီးပါးမှာ အခြေခံသည်ကို တွေ့နိုင်ပါသည်။ Linear Algebra အသုံးဝင်ပုံသည် နှစ်ပိုင်း ရှိပါသည်။

- သီအိုရီအားဖြင့် Linear Algebra သည် Space, Dimension, Transformation, Dependency များကို လေ့လာရာမှာ အင်မတန် အသုံးဝင်ပါသည်။ နောက်ပိုင်း Quantum Physics တွင် Tensor Algebra လို့ ခေါ်တဲ့ Advanced Linear Algebra များကို မသုံးမဖြစ် သုံးရပါသည်။ ဥပမာ၊ Quantum Superposition၊ Quantum Entanglement တို့သည် Space (Hilbert Space) တွေပေါ်မှာ အများကြီးမူတည်ပါသည်။

- လက်တွေ့အားဖြင့် Linear Algebra သည် ပိုလို့တောင် အရေးကြီးပါသည်။ Excel Spreadsheet သည် တကယ်တော့ Matrix တစ်ခု ဖြစ်သည်။ SQL Table များသည်လည်း Matrix ဖြစ်သည်။ ပြီးတော့ ကျွန်တော်တို့ Game ကစားရင် GPU က တွက်တဲ့ Graphics Calculation များ၊ Machine Learning မှာ သုံးမည့် Dataset များသည် Vector, Matrix နှင့် Tensor များဖြစ်ကြသည်။ ဒါကြောင့် Google က Tensorflow လို့ နာမည်ပေးတာ ဖြစ်သည်။

- တန်ဖိုးများကို အတွဲလိုက် ဖော်ပြရုံဖြင့် ကျွန်တော်တို့ မသိသေးတဲ့ Space တွေကို စဉ်းစားလို့ ရလာတာသည် အင်မတန် စိတ်ဝင်စားဖို့ ကောင်းသည်။ ဥပမာ၊ စာရွက်အလွတ်နှင့် ခဲတံရှိရုံဖြင့် စာရွက်ပေါ်မှာ ဆွဲနိုင်၊ ရေးနိုင်တာတွေသည် Infinitely Possible ဖြစ်သည်။ Mona Lisa ပုံဖြစ်မည်၊ ဂျပန်စာဖြစ်မည်၊ သင်္ချာ ဖော်မြူလာဖြစ်မည်။ ထိုစာရွက်သည် ထို Infinite Possibility ၏ Space တစ်ခု ဖြစ်သည်။ ကျွန်တော်မြင်တာက Space တစ်ခုသည် Boundary နှင့် Possibility ကို ဖော်ပြသည်လို့ မြင်ပါသည်။ (ကျွန်တော့်အမြင်သာ ဖြစ်သည်။)

- ဒါကြောင့် Linear Algebra ရဲ့ Advanced Topics များကို ဆက်ပြီးဆွေးနွေးပါမည်။

Vector Space

- Vector များကို ဖော်ပြလို့ ရတဲ့ N Dimensional Space တစ်ခုကို Vector Space ဟုခေါ်ပါသည်လို့ ကျွန်တော်တို့ အကြမ်းဖျဉ်း ပြောခဲ့ပါသည်။ အခု ဒါကို ပိုပြီးတော့ သေသေချာချာ ပြောဖို့လိုပါသည်။ Vector Space သည် တော်တော်လေး စိတ်ဝင်စားဖို့ ကောင်းပါသည်။

- V က Vector Space တစ်ခုဆိုပါတော့။ ကျွန်တော်တို့က Closure ကို ပထမ Define လုပ်ပါသည်။

```
Additive Closure                     : v + u ∈ V
Scalar Multiplicative Closure        : av ∈ V , a = scalar
```

- ထို့နောက် ကျွန်တော်တို့က Axioms များကို Define လုပ်ပါသည်။

$$
\begin{aligned}
&(\mathbf{u} + \mathbf{v}) + \mathbf{w} = \mathbf{u} + (\mathbf{v} + \mathbf{w}) \\
&\mathbf{u} + \mathbf{0} = \mathbf{0} + \mathbf{u} = \mathbf{u} \\
&\mathbf{u} + (-\mathbf{u}) = (-\mathbf{u}) + \mathbf{u} = \mathbf{0} \\
&\mathbf{v} + \mathbf{u} = \mathbf{u} + \mathbf{v} \\
&k(\mathbf{u} + \mathbf{v}) = k\mathbf{u} + k\mathbf{v} \\
&(a + b)\mathbf{u} = a\mathbf{u} + b\mathbf{u} \\
&(ab)\mathbf{u} = a(b\mathbf{u}) \\
&\mathbf{1u} = \mathbf{u}
\end{aligned}
$$

- Closure နှင့် Axioms များကို ပြေလည်သော အရာမှန်သမျှကို Vector Space လို့ခေါ်နိုင်သည်။ ကျွန်တော်တို့ သိသမျှ Function တွေ၊ Derivative တွေ၊ Integral တွေ၊ Matrix တွေ၊ Tensor တွေ၊ Polynomial တွေသည် Vector Space များဖြစ်သည်။

- ဒီဟာကို သေသေချာချာလေး စဉ်းစားစေချင်ပါသည်။

Sub Space

- V က Vector Space တစ်ခုဆိုပါတော့။ W က V ရဲ့ Subset တစ်ခု ဆိုပါက W သည် V ၏ Sub Space တစ်ခု ဖြစ်ပြီး V ၏ Closure နှင့် Axioms များကို လိုက်နာသည်။

$$P(x) = a_1x + a_2x^2 + \dots + a_nx^n$$

- P(x) သည် Polynomial ဖြစ်ပြီး Vector Space တစ်ခု ဖြစ်သည်ဆိုပါက

$$Q(x) = a_1x + a_2x^2$$

- Q(x) သည် P(x) ၏ Sub Space တစ်ခု ဖြစ်သည်။

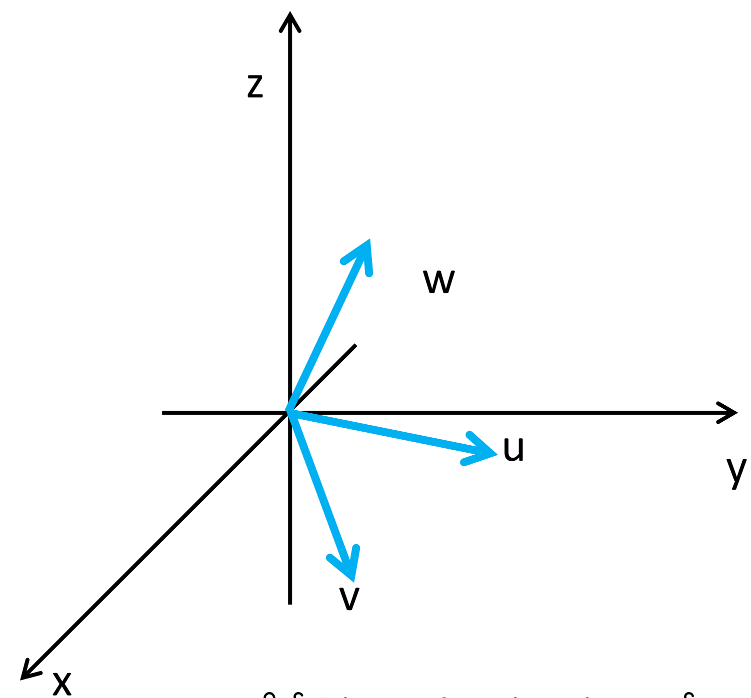
Linear Span, Basis and Dimension

- **V** က Vector Space တစ်ခုဆိုပါတော့။ Vector Space အတွင်းရှိ Vector များကို Linear Combination လုပ်ခြင်းဖြင့် Linear Span ကို ရရှိပါသည်။ Linear Span သည် Additive Closure နှင့် Scalar Multiplicative Closure ကို အခြေခံခြင်း ဖြစ်သည်။ ဆိုလိုသည်က Vector Space အတွင်းရှိ Vector များ၏ ကြိုက်တဲ့ Linear Combination သည် ထို Vector Space အတွင်းမှာပဲ အမြဲ ရှိပါသည်။

- တကယ်လို့ Vector Space တစ်ခု၏ Linear Span သည် Linearly Independent ဖြစ်သွားပြီး ထို Vector Space ကို ကိုယ်စားပြု (Span) ဖြစ်သွားပြီဆိုရင် ထို Linear Span ကို ထို Vector Space ၏ Basis ဟု ခေါ်သည်။

- Basis တစ်ခု၏ Element အရေအတွက်ဟု ခေါ်သည်။ Basis ၏ အရေအတွက်က 3 ဖြစ်လျှင် ထို Basis က Span လုပ်သော Vector Space ကို 3 Dimensional Vector Space ဟုခေါ်သည်။ Dimension များသည် Infinite ဖြစ်နိုင်ပါသည်။

- Basis Vector များ၏ Linear Span ရှိ Scalar Multiplier များကို Coordinates ဟုခေါ်သည်။



u, v, w တို့၏ Linear Combination သည် Linear Span ဖြစ်ပြီး **x, y, z** (Basis) Vector Space ထဲမှာ ရှိမည်။

u, v, w တို့သည် Linearly Independent ဖြစ်သွားလျှင် **x, y, z** Vector Space ကို ကိုယ်စားပြုသော Basis များ ဖြစ်သွားမည်။

$\mathbf{r} = a\mathbf{x} + b\mathbf{y} + c\mathbf{z}$ တွင် a, b, c တို့သည် Coordinates များဖြစ်သည်။

Inner Product Space

- **V** က Vector Space တစ်ခုဆိုပါတော့။ **v, u** တို့သည် Vector Space အတွင်းရှိ Vector ဖြစ်ပြီး **<u, v>** သည် **v, u** Pair ကို Real Number တစ်ခု အပေါ် Assign လုပ်တဲ့ Function တစ်ခု ဆိုပါက အောက်ပါအချက်များကို လိုက်နာရင် **<u, v>** ကို Inner Product ဟု ခေါ်ပြီး **V** ကို Inner Product Space ဟုခေါ်သည်။

```
Linearity                : <av + bu, w> = a<v, w> + b<u, w>
Symmetry                 : <u, v> = <v, u>
Positive Definite        : <u, u> >= 0
```

- Dot Product သည် Inner Product Space ထဲရှိ Function တစ်ခု ဖြစ်သည်။ သို့သော် Matrix Multiplication သည် Symmetry ကို မပြေလည်သည့်အတွက် Inner Product Space ထဲရှိ Function တစ်ခု မဟုတ်ပါ။

- Hilbert Space သည် Inner Product Space တစ်ခုဖြစ်ပြီး အခြား Properties များလည်းရှိပါသည်။

Row and Column Space of a Matrix

- Matrix တစ်ခု၏ Rows သို့မဟုတ် Columns များကို Vector များအနေဖြင့် ယူဆနိုင်သည်လို့ ပြောခဲ့ပါသည်။ ထို Row Vector များ နှင့် Column Vector များက Linear Span လုပ်သော Vector Space ကို Row Space သို့မဟုတ် Column Space ဟုခေါ်ပါသည်။

- Row Space သို့မဟုတ် Column Space က Bound လုပ်ထားတဲ့ Region ရဲ့ Volume သည် ထို Matrix ၏ Determinant ဖြစ်သည်။

- Row Space of Matrix A နှင့် Column Space of Matrix B တို့မြှောက်ခြင်းသည် Matrix Multiplication (Inner Product) ဖြစ်သည်။ တစ်နည်း Row Space of Matrix A နှင့် Column Space of Matrix B တို့၏ Linear Combination သည် Matrix Multiplication ဖြစ်သည်။

- Column Space of Matrix A နှင့် Row Space of Matrix B တို့မြှောက်ခြင်းသည် Outer (Tensor) Product ဖြစ်သည်။ တစ်နည်း Column Space of Matrix A နှင့် Row Space of Matrix B တို့၏ Linear Combination သည် Outer (Tensor) Product ဖြစ်သည်။

System of Equations

$$
\begin{array}{l}
a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\
a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\
\qquad\qquad \dots \\
a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_n
\end{array}
$$

$$
\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}
$$

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

- Systems of Equations များသည် အမှန်အားဖြင့် Zero Solution (အဖြေမရှိ)၊ Unique Solution (အဖြေတစ်ခုရှိ) သို့မဟုတ် Infinite Solutions (မရေနိုင်သော အဖြေရှိ) ဆိုပြီး ဖြစ်နိုင်ပါသည်။

- Row Dimension (m) သည် Equation အရေအတွက်ကို ပြောတာဖြစ်ပြီး Column Dimension (n) သည် Unknown (x_n) အရေအတွက်ကို ပြောတာဖြစ်သည်။

- ဒါကြောင့် ကျွန်တော် Systems of Equations များ အဖြေရှိ၊ မရှိ။ ပြီး အဖြေဘယ်လို ရှာမလဲကို စိတ်ဝင်စားပါသည်

- Augmented Matrix M သည် Coefficient Matrix A နှင့် Constant Vector **b** တို့ကို တွဲထားတဲ့ Matrix ကို ခေါ်ပါသည်။

$$
\begin{pmatrix} a_{11} & \cdots & a_{1n} & b_1 \\ \vdots & \ddots & \vdots & \vdots \\ a_{m1} & \cdots & a_{mn} & b_n \end{pmatrix}
$$

Uniqueness and Existence Theorem

$$M = \begin{pmatrix} a_{11} & \cdots & a_{1n} & b_1 \\ \vdots & \ddots & \vdots & \vdots \\ a_{m1} & \cdots & a_{mn} & b_n \end{pmatrix} \qquad A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}$$

- Uniqueness and Existence Theorem အရ
  - Rank (A) နှင့် Rank (M) တို့တူမှသာ ထို Systems of Equations များအတွက် Solution ရှိမည် ဖြစ်သည်။
  - Rank (A) နှင့် Rank (M) တို့သည် A ရဲ့ Column Dimension (n) နှင့် တူလျှင် Unique Solution ရှိမည် ဖြစ်သည်။

- တကယ်လို့ ထို Systems of Equations များ Solution တစ်ခုထက် ပိုရှိလျှင် Infinite Number of Solutions ရှိသည်ဟု ဆိုနိုင်ပါသည်။

Triangular Matrix

$$U = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{pmatrix} \qquad L = \begin{pmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

[Triangular Matrix - Wikipedia]

- Diagonal Line ရဲ့ အောက်မှာ zero သာရှိရင် Upper Triangular Matrix ဟုခေါ်ပြီး Diagonal Line ရဲ့ အပေါ်မှာ zero သာရှိရင် Lower Triangular Matrix ဟုခေါ်သည်။

- Triangular Matrix များသည် Numerical Solutions များအတွက် အင်မတန် အသုံးဝင်ပါသည်။

- တကယ်လို့သာ Systems of Equations များ၏ Coefficient Matrix A သည် Triangular Matrix သာဆိုရင် Equation များကို လွယ်လွယ်ကူကူ ရှင်းလို့ ရမည်။

Gaussian Elimination

$$M = \begin{pmatrix} a_{11} & \cdots & a_{1n} & b_1 \\ \vdots & \ddots & \vdots & \vdots \\ a_{m1} & \cdots & a_{mn} & b_n \end{pmatrix} \qquad A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}$$

[Gaussian Elimination - Wikipedia]

- Systems of Equation များရဲ့ အဖြေရှာဖို့ကို Gauss က တင်ပြခဲ့သည့်အတွက် ဒီ Method ကို Gaussian Elimination ဟုခေါ်ပါသည်။
- Gaussian Elimination မှာ 2 ပိုင်းပါပါသည်။
  - Augmented Matrix M ကို Triangular Matrix သို့ ပြောင်းပါသည်။ တဆင့်ပြီး ပြောင်းရတဲ့ Step တွေကို Row Operations ဟုခေါ်ပါသည်။ ရလာတဲ့ Triangular Matrix ကို Row Echelon Form ခေါ်ပါသည်။
  - ရလာတဲ့ Triangular Matrix ကိုသုံးပြီး Value များကို Substitute လုပ်ခြင်းဖြင့် အဖြေရှာပါသည်။

Gaussian Elimination

To find Triangular Form

To substitute

System of equations	Row operations	Augmented matrix
$\begin{alignedat}{4} 2x &+ y &- z &= 8 \\ -3x &- y &+ 2z &= -11 \\ -2x &+ y &+ 2z &= -3 \end{alignedat}$		$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ -3 & -1 & 2 & -11 \\ -2 & 1 & 2 & -3 \end{array}\right]$
$\begin{alignedat}{4} 2x &+ y &- z &= 8 \\ & \tfrac{1}{2}y &+ \tfrac{1}{2}z &= 1 \\ & 2y &+ z &= 5 \end{alignedat}$	$L_2 + \tfrac{3}{2}L_1 \to L_2$ $L_3 + L_1 \to L_3$	$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ 0 & \tfrac{1}{2} & \tfrac{1}{2} & 1 \\ 0 & 2 & 1 & 5 \end{array}\right]$
$\begin{alignedat}{4} 2x &+ y &- z &= 8 \\ & \tfrac{1}{2}y &+ \tfrac{1}{2}z &= 1 \\ & & -z &= 1 \end{alignedat}$	$L_3 + -4L_2 \to L_3$	$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ 0 & \tfrac{1}{2} & \tfrac{1}{2} & 1 \\ 0 & 0 & -1 & 1 \end{array}\right]$
The matrix is now in echelon form (also called triangular form)		
$\begin{alignedat}{4} 2x &+ y & &= 7 \\ & \tfrac{1}{2}y & &= \tfrac{3}{2} \\ & & -z &= 1 \end{alignedat}$	$L_2 + \tfrac{1}{2}L_3 \to L_2$ $L_1 - L_3 \to L_1$	$\left[\begin{array}{ccc\|c} 2 & 1 & 0 & 7 \\ 0 & \tfrac{1}{2} & 0 & \tfrac{3}{2} \\ 0 & 0 & -1 & 1 \end{array}\right]$
$\begin{alignedat}{4} 2x &+ y & &= 7 \\ & y & &= 3 \\ & & z &= -1 \end{alignedat}$	$2L_2 \to L_2$ $-L_3 \to L_3$	$\left[\begin{array}{ccc\|c} 2 & 1 & 0 & 7 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -1 \end{array}\right]$
$\begin{alignedat}{4} x & & &= 2 \\ & y & &= 3 \\ & & z &= -1 \end{alignedat}$	$L_1 - L_2 \to L_1$ $\tfrac{1}{2}L_1 \to L_1$	$\left[\begin{array}{ccc\|c} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -1 \end{array}\right]$

LU Decomposition

A = $\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$ U = $\begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix}$ L = $\begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix}$

```
A = LU
```

LU Decomposition - Wikipedia

- Non Singular Matrix A တစ်ခုကို Upper Triangle Matrix U နှင့် Lower Triangle Matrix L တို့၏ မြှောက်လဒ်အနေဖြင့် ဖော်ပြလို့ ရပါသည်။

Homogenous Equations

$$A\mathbf{x} = \mathbf{b}$$

- Systems of Equations များကို ကျွန်တော်တို့ ဒီလို ဖော်ပြလို့ ရပါသည်။ တကယ်လို့ **b** သည် Zero ဖြစ်သွားမည်ဆိုလျှင် Homogenous Equation ဟုခေါ်ပြီး အနည်းဆုံးတော့ **x** = 0 Solution (Trivial Solution) ရှိမည် ဖြစ်သည်။

$$A\mathbf{x_h} = \mathbf{0}$$

$$A\mathbf{x}_p = \mathbf{b}$$

$$\mathbf{A(x_h) + A(x_p)} = \mathbf{b}$$

- Inhomogeneous Equations များ၏ Solution သည် Homogenous Equation ၏ Solution အားလုံးကို Particular Solution $A\mathbf{x}_p = \mathbf{b}$ ကို ထပ်ပေါင်းထားခြင်းသာ ဖြစ်သည်။

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$$

- Homogenous Equation တစ်ခု၏ m သည် Number of Equations ဖြစ်ပြီး n သည် Number of Unknowns ဆိုပါက
  - m = n ဆိုလျှင် Trivial Solution တစ်ခုသာရှိမည်။
  - m < n ဆိုလျှင် Infinite Solutions ရှိမည်။

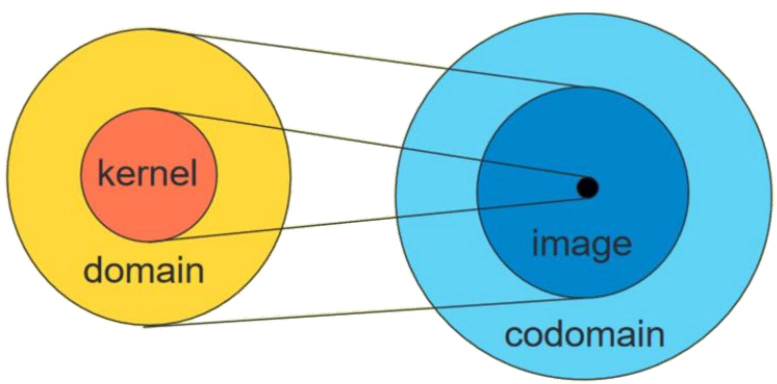
Kernel and Image

$$A\mathbf{x} = \mathbf{b}$$

- Systems of Equations များကို Linear Transformation လို့ ယူဆပါက b ကို A ၏ Image တစ်ခုဖြစ်သည် လို့ပြောပါသည်။

$$A\mathbf{x} = \mathbf{0}$$

- Homogeneous Systems of Equations များကို Linear Transformation ယူဆပါက A ကို Linear Mapping to a Null Space ဟုခေါ်ပြီး $A\mathbf{x} = \mathbf{0}$ ဖြစ်စေသော $\mathbf{x}$ တန်ဖိုးများကို Kernel ဟုခေါ်သည်။



Eigen Vectors and Values

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

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \lambda \begin{pmatrix} x_1 \\ \vdots \\ x \end{pmatrix}$$

- Systems of Equation များ သို့မဟုတ် Linear Mapping တစ်ခုသည် Unknown Vector **x** ရဲ့ Scalar **λ** Multiplication တစ်ခုသာ ဆိုရင် **x** ကို Eigen Vector လို့ခေါ်ပြီး **λ** ကို Eigen Value လို့ခေါ်သည်။

- Eigen Vectors and Values သည်လည်း အင်မတန် အသုံးဝင်ပါသည်။

$$\begin{aligned} A\mathbf{x} &= \lambda\mathbf{x} \\ A\mathbf{x} - \lambda\mathbf{x} &= 0 \\ (A - \lambda I)\mathbf{x} &= 0 \end{aligned}$$

Since x cannot be zero, $(A - \lambda I)$ = Singular Matrix

$$\det(A - \lambda I) = 0$$

Characteristics of Eigen Equation

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

- Eigen Equation သည် အင်မတန် ရိုးရှင်းသော်လည်း အင်မတန် လှပ လေးနက်ပါသည်။

$$\frac{d}{dx}e^{x} = e^{x}$$

$$\frac{d}{dx^{2}}\sin(x) = -\sin(x)$$

$$\frac{d}{dx^{2}}\cos(x) = -\cos(x)$$

- Derivative of Exponential Function သည် Eigen Equation တစ်ခု ဖြစ်သည်။ Second Derivative of Sin သို့မဟုတ် Cos Function သည်လည်း Eigen Equation တစ်ခု ဖြစ်သည်။

- နောက်ပိုင်းမှာ Principle Component Analysis လုပ်မည်ဆိုပါက Covariance Matrix များ၏ Eigen Vector များသည် Principle Component များဖြစ်ကြသည်။

Linear Operator

- Differential များ၊ Integral များသည် Vector Space ထဲမှာ ပါသည်ဟု ပြောခဲ့ပါသည်။ ထို့ကြောင့် Differential များ၊ Integral များ Linear Mapping များ အနေဖြင့် ဖော်ပြလို့ ရပါသည်။ ဒီလို ဖော်ပြခြင်းကို Linear Operator ဟုခေါ်ပါသည်။

- ထိုအထဲမှ
  - Hessian Matrix (Second Order Scalar Derivative)

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

  - Jacobian Matrix (First Order Vector Derivative)

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_m}{\partial x_1} & \cdots & \dfrac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Applications of Linear Algebra

- Computer များသည် အမှန်တော့ ဒုတိယကမ္ဘာစစ်အတွင်းနှင့် စစ်ပြီးမှ အသုံးများလာခြင်း ဖြစ်သည်။

- စစ်အတွင်းမှာ Logistics နှင့် Decision Analysis တွေလုပ်ဖို့ သင်္ချာတွေ အများကြီးသုံးပါသည်။ ဒါကို လေ့လာဖို့ Operations Research ဆိုပြီးတော့တောင် နယ်ပယ်တစ်ခု ထွက်ပေါ်လာခဲ့ပါသည်။ အခုတော့ ဒါကို Management Science ဟုခေါ်ပါသည်။ အဓိကက Optimization လုပ်ဖို့နှင့် Decision Analysis and Support ဖြစ်သည်။

- Systems of Equation များသည် အမှန်တော့ Decision Analysis လုပ်ဖို့ အတွက်သုံးလို့ ရပါသည်။ Computer များအသုံးမတွင်ခင်က လက်ဖြင့်တွက်ရတာဖြစ်သည်။ သို့သော် ဒါကို Computer များပေါ်မှာ Program ရေးလို့ရသည့်အခါ အများကြီး အချိန်ကုန်သက်သာသွားသည်။ စစ်အတွင်းမှာ ဒီအချက်က အင်မတန်အရေးကြီးပါသည်။

- နောက်ပိုင်းမှာ Computer Graphics ဆိုပြီး ပေါ်လာပါသည်။ ဒီနယ်ပယ်ကို Pixar Founder Edwin Catmull ရဲ့ဆရာ Ivan Edward Sutherland တို့က စတာဖြစ်သည်။ Edwin Catmull လဲ Pioneer ထဲမှာပါပါသည်။ Computer Graphics တွေကို တွက်ဖို့ Special CPU တစ်ခု သပ်သပ်လိုပါသည်။ ဘာဖြစ်လို့လဲ ဆိုတော့ Homogenous Coordinates မှာပြောခဲ့တဲ့ 3D Graphics တွေကို တွက်ဖို့၊ Render လုပ်ဖို့သည် Matrix and Vector Calculation များကို တွက်ဖို့လိုပါသည်။ ဒါနဲ့ GPU များပေါ်လာတာဖြစ်သည်။ Steve Jobs က Pixar တည်ထောင်ပြီး 3D Graphics တွေတွက်တဲ့ Computer များကို ရောင်းဖို့ဖြစ်သည်။

- GPU များပေါ်မှာ တွက်တဲ့ Computer Instruction များသည် SIMD (Single Instruction Multiple Data) ဖြစ်သည်။ GPU များပေါ်မှာ တွက်တဲ့ Processing Model ကို Stream Processing Model လို့ခေါ်ပါသည်။

- ထို့အတူ Machine Learning မှာ တွက်ရ၊ ချက်ရသည်က Tensor များဖြစ်သည်။ ဒါကြောင့် GPU ကို သုံးနိုင်းတာဖြစ်သည်။

- ဒါဆိုရင် ဘာကြောင့် Linear Algebra သည် Machine Learning မှာ အရမ်းအရေးကြီးတယ်ဆိုတာ သိလောက်ပါပြီ။

Summary

- အခု Space ဆိုတာကို နားလည် သလိုလိုတော့ ရှိသွားသည် ထင်ပါသည်။

- အမှန်တော့ ကျွန်တော်ကိုယ်တိုင်လဲ နားမလည်သေးပါ။

- သို့သော် ကျွန်တော်နားလည်သည်က Space တစ်ခုသည် Boundary နှင့် Possibility ကို ဖော်ပြတာ ဖြစ်နိုင်ပါသည်။

- Matrix သည် တကယ်တော့ ဇယားကွက်ကလေး တစ်ခုသာ ဖြစ်ပါသည်။ သို့သော် ထို ဇယားကွက်ကလေး ထဲမှာ ဖြစ်နိုင်သော၊ ထို ဇယားကွက်ကလေးက ဖော်ပြနိုင်သော Possibility များသည် Infinite ဖြစ်သည်။ ဥပမာ၊ Excel Sheet မှာ ဖော်ပြနိုင်တာသည် အကြွေးစာရင်းဖြစ်မည်၊ အရောင်းစာရင်းဖြစ်သည်၊ နာမည်များဖြစ်မည်။ အခု ကမ္ဘာပေါ်ရှိ Excel Sheet အားလုံးမှာ သိမ်းထားသော Number များသည် သူ့အစီပွါယ်နှင့် သူရှိနေမည်ဖြစ်သည်။

- ထို့ကြောင့် Space များသည် နားမလည်နိုင်သော်လည်း စိတ်ဝင်စားဖို့ ကောင်းပါသည်။