IN2070 – høst 2023

Fourier I -- En litt annen vinkling på stoffet i kapittel 4

I dag:

- Sinus-funksjoner i 1D og 2D
- 2D diskret Fouriertransform (DFT)

Neste uke:

- Konvolusjonsteoremet
- Filtre og filtrering i frekvensdomenet
- Vindusfunksjoner

Introduksjon I/II

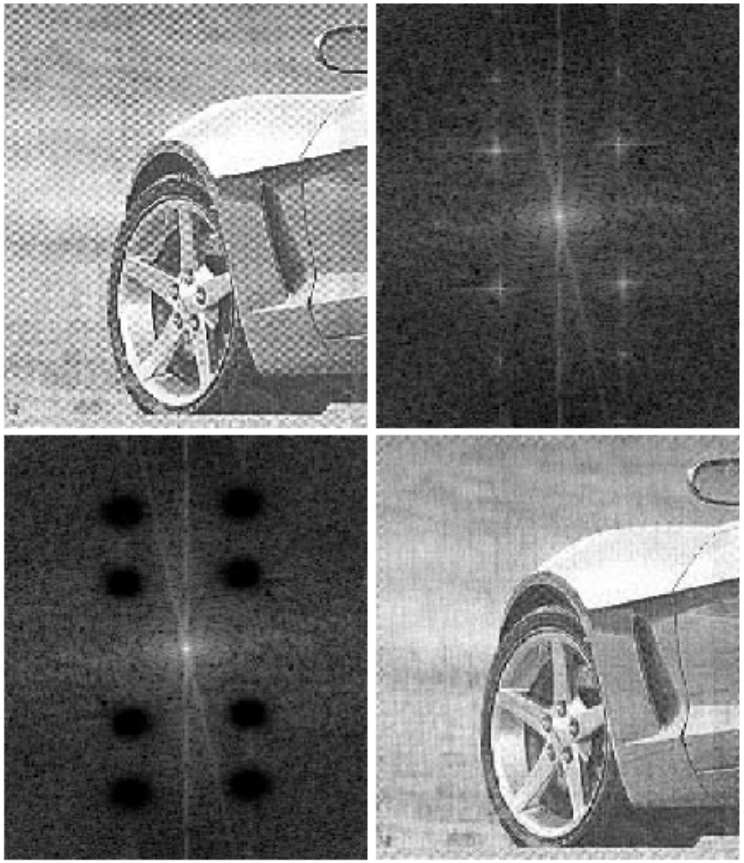
- Et gråtonebilde
  - Typisk representasjon: Matrise av gråtoneintensiteter
  - Fourier: En vektet sum av sinuser og cosinuser med ulik frekvens og orientering



- Et slikt skifte av representasjon kalles ofte et «basis-skifte»

Introduksjon II/II

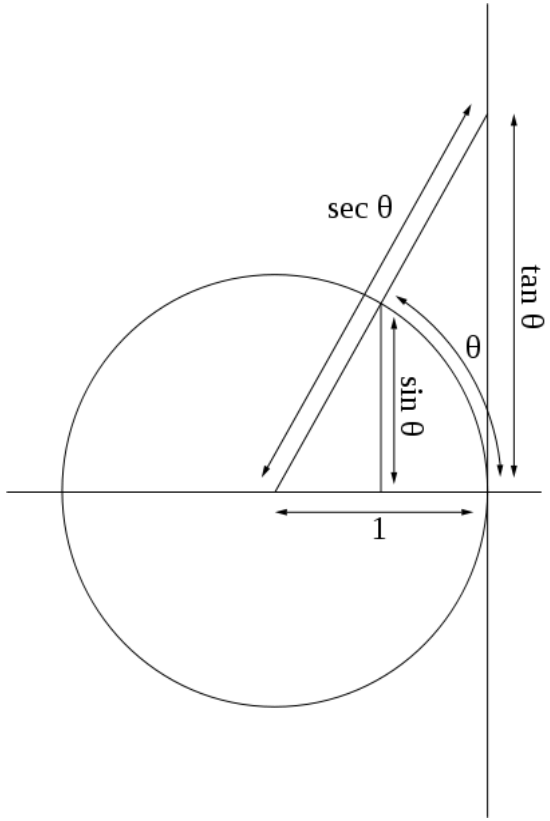
- Hvorfor skifte basis?
  - Analyse (og syntese) av bilder
    - Skarphet, om er orienteringsdominert, etc.
    - Fjerne/dempe periodisk støy
    - Kompresjon
  - Analyse og design av lineære filtre (konvolusjonsteoremet)
  - Egenskapsuttrekning (feks. Tekstur)
  - Rask implementasjon av (større) konvolusjonsfiltre
  - Fordi vi kan!

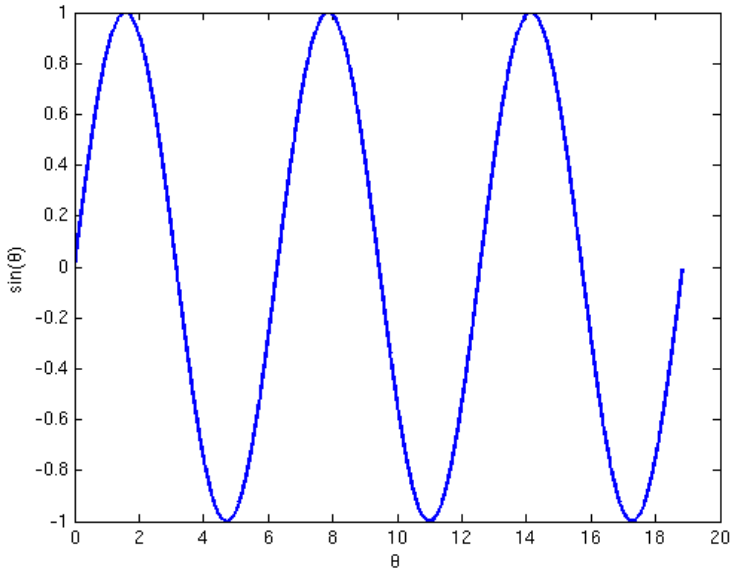


Fjerning av periodisk støy, fig. 4.64 i DIP. Ut-bildet er resultatet av en konvolusjon, men det er vanskelig å designe filteret i bildedomenet.

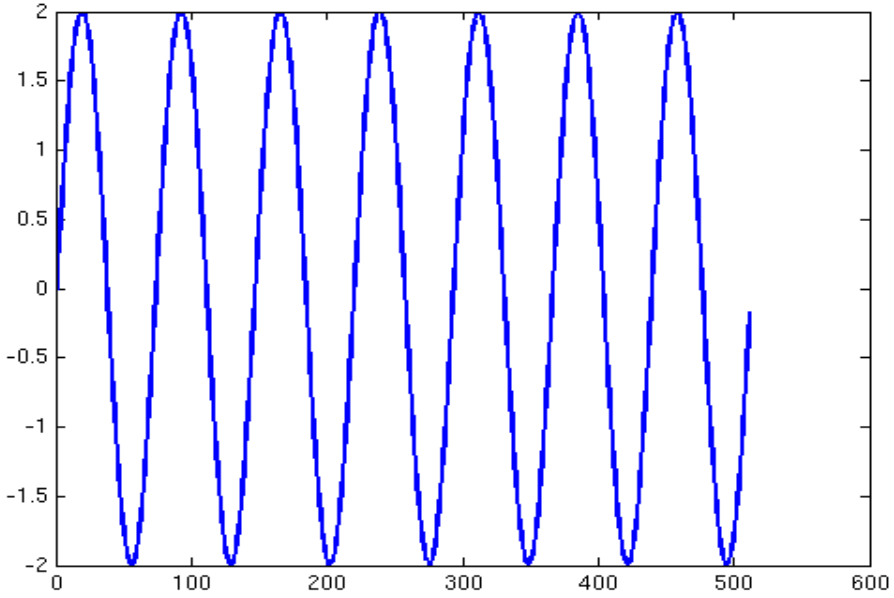
Funksjonen sin(θ)

sin(θ) svinger mellom 1 og -1 når θ varierer mellom 0 og 2π, og den svinger på samme måte når θ varierer mellom 2π og 4π osv. (periodisk)





«Diskret» sin i 1D



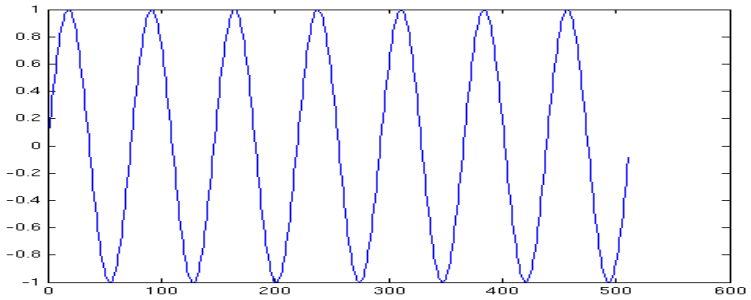
$$y(i) = A \sin(2\pi ui/N + \varphi)$$

N : antall sampler
u : antall hele perioder
φ : horisontal forskyvning (fase)
A : Amplitude

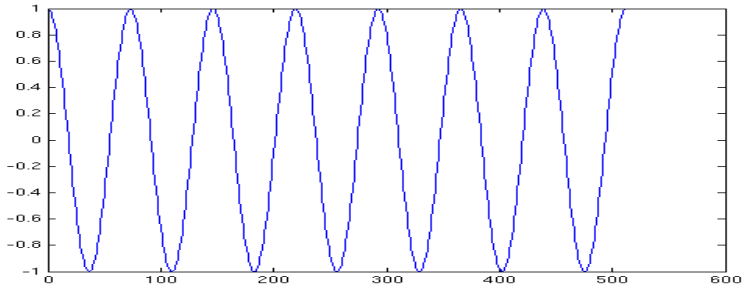
I dette eksemplet er
A=2, u=7, N=512 og ϕ=0

Hva er forskjellen på sin(θ) og cos(θ)?

- **sin(2πui/N)** starter på 0 og repeteres u ganger per N samples



- **cos(2πui/N)** starter på 1 og repeteres u ganger per N samples



Bare startpunktet, dvs. faseforskyvningen, ϕ, er forskjellig (90°)

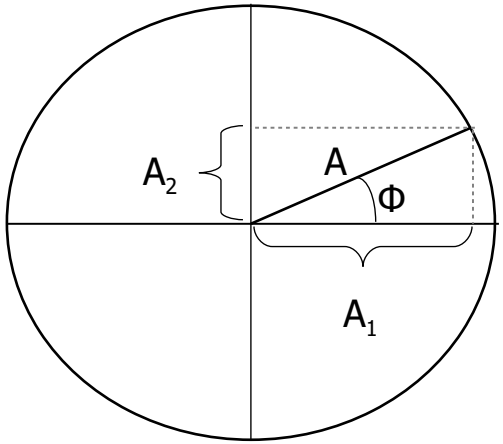
Hva får vi om vi legger sammen sin og cos?

- $A_1\sin(\Delta i) + A_2\cos(\Delta i) = A\sin(\Delta i + \Phi)$,

  der $A = \sqrt{A_1^2 + A_2^2}$ og $\Phi = \text{atan2}(A_2, A_1)$

- Vi ender altså opp med en sin-funksjon med samme frekvens, men endret amplitude og fase

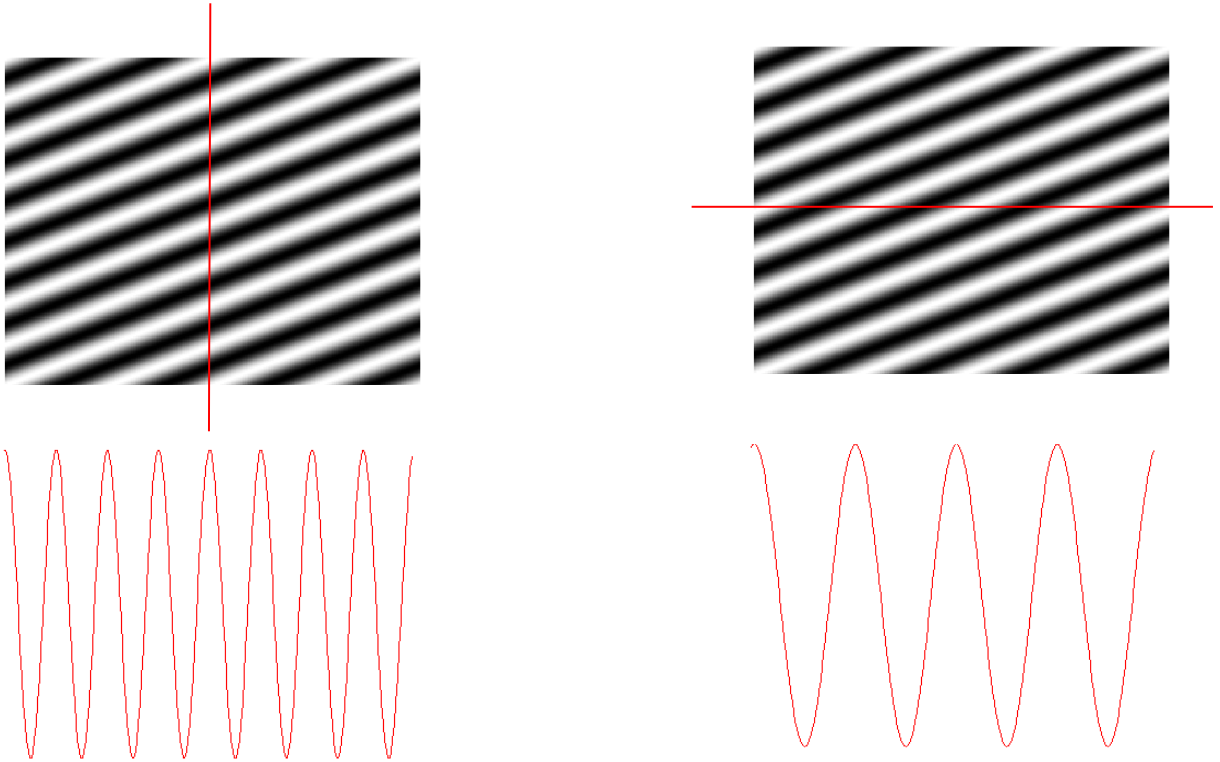
- Vi kan også gå andre veien og si at **enhver sinus-funksjon** med gitt frekvens kan dannes ved å **legge sammen en vektet sin- og en vektet cos-funksjon** med denne samme frekvensen



Alternativ "koding"/representasjon av informasjonen (A, Φ, Δ) er altså (A_1, A_2,Δ)

Introduksjon til sinus-funksjoner i 2D

- Vertikal og horisontal komponent



«Diskret» sinus/cosinus i 2D

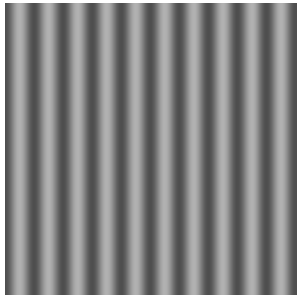
$$f(x,y) = A\sin(\frac{2\pi(ux+vy)}{N} + \phi)$$

A – amplitude

u – vertikal frekvens

v – horisontal frekvens

ϕ - faseforskyvning

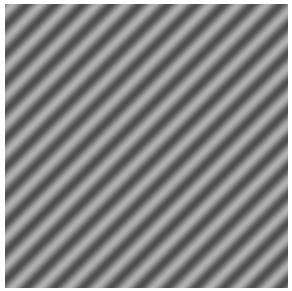


A=50, u=0, v=10

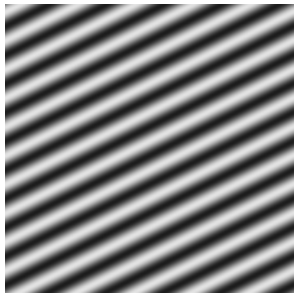


A=20, u=10, v=0

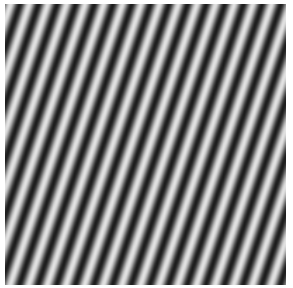
I eksemplene vises 0 som grått,
-127 som sort, og 127 som hvitt



A=50, u=10, v=10



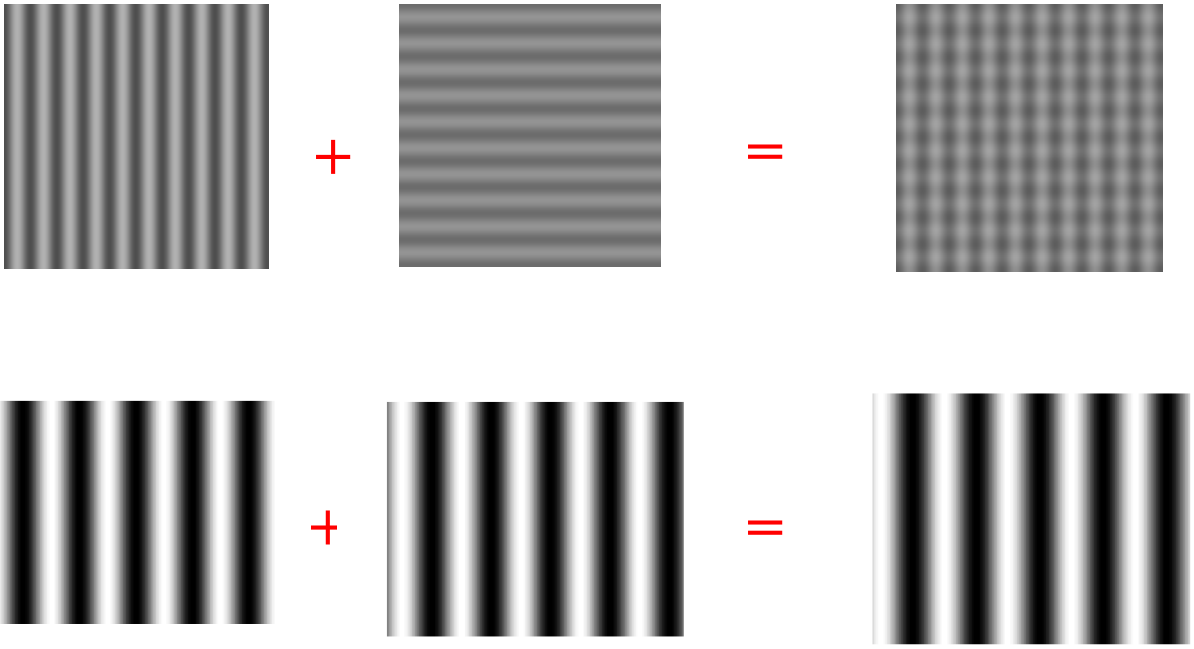
A=100, u=10, v=5



A=100, u=5, v=15

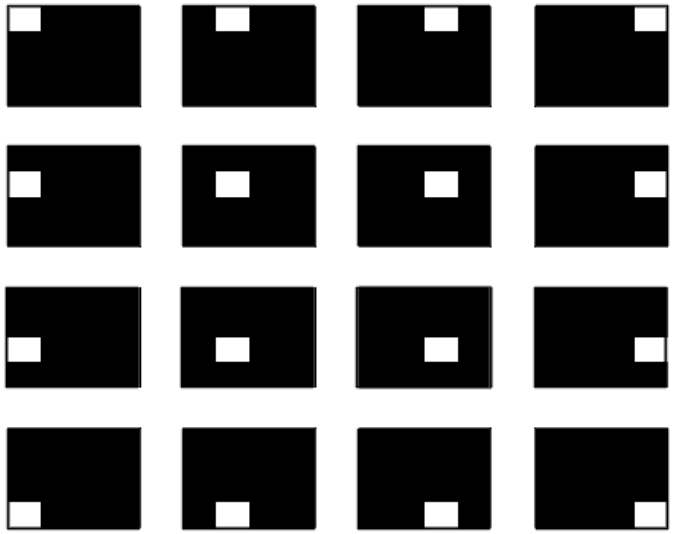
Merk: u og v er antall repetisjoner i bildet vertikalt og horisontalt

Eksempler: Sum av 2D sinfunksjoner



Sum av to bilder med lik frekvens (og lik retning) gir nytt bilde med samme frekvens (og retning), jfr s. 7.

«Basis-bilder»



Sort er 0, hvit er 1.

Ortogonal basis for alle 4x4 gråtonebilder.

Eksempel:

1	3	2	1
5	4	5	3
4	1	1	2
2	3	2	6

$= 1*$  $+ 3*$  $+\ldots+ 6*$ 

Alternativ basis (Fourier)

- Bildene

$$\cos(\frac{2\pi(ux+vy)}{N}) \qquad \sin(\frac{-2\pi(ux+vy)}{N})$$

- med frekvensene

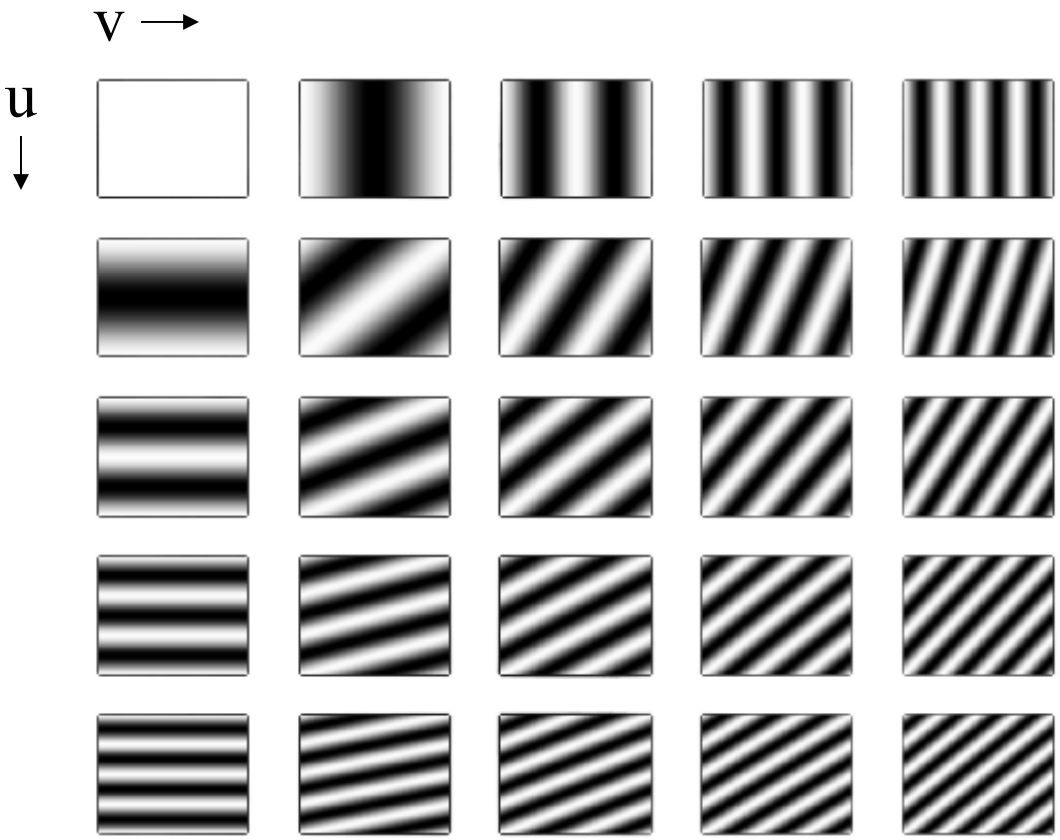
$$u = 0,1,\ldots,N\text{-}1$$
$$v = 0,1,\ldots,N\text{-}1$$

- Alle digitale gråtonebilder av størrelse NxN kan representeres ved en vektet summasjon av disse NxN sinus- og cosinus-bildene (basisbilder/basisvektorer)

Denne basisen er også **ortogonal**, sett bort i fra duplikat-komponentene grunnet symmetriene og antisymmetriene til cos og sin (noe vi kommer til om litt)

Ved ikke-kvadratiske bilder:
cos(2π(ux/M+vy/N))
sin(-2π(ux/M+vy/N))

Basisbilder - cosinus



… til v = N-1

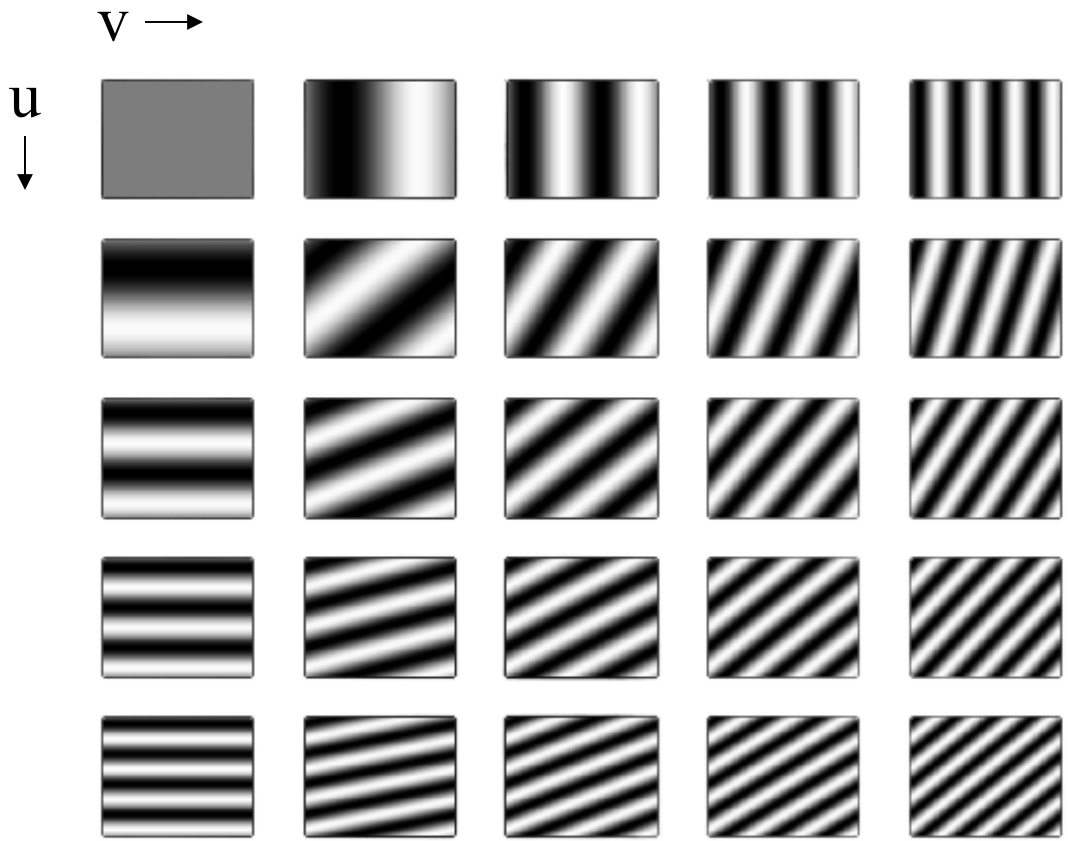
.

.

til u = N-1

I illustrasjonen indikerer sort -1 og hvitt 1

Basisbilder - sinus



… til v = N-1

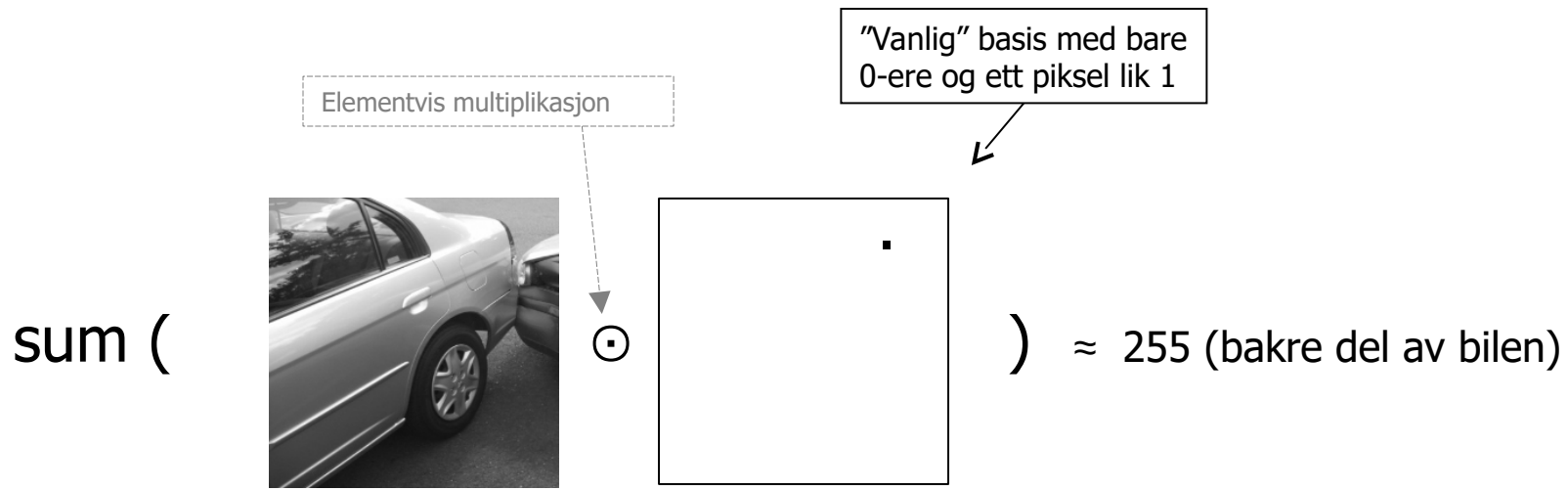
.

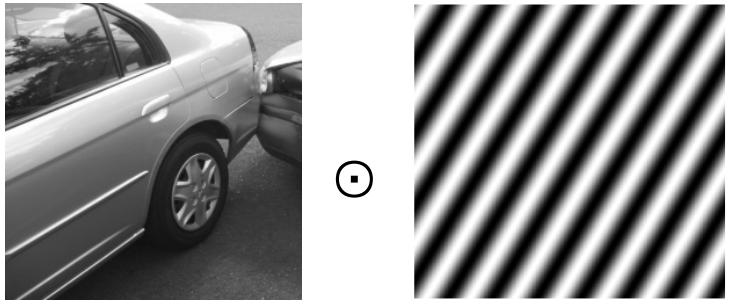
.

til u = N-1

I illustrasjonen indikerer sort -1 og hvitt 1

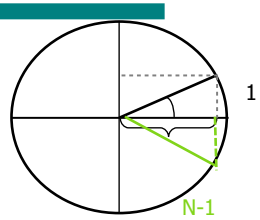
Hvordan finne bidraget fra et gitt basisbilde?



- sum ( ⊙)

Bidraget finnes altså ved **indreproduktet** mellom bildet og basisbildet

Symmetri/duplikater i basisbildene



cosinus



v= 0 1 2 3 4 …. N-5 N-4 N-3 N-2 N-1

u= 0

sinus



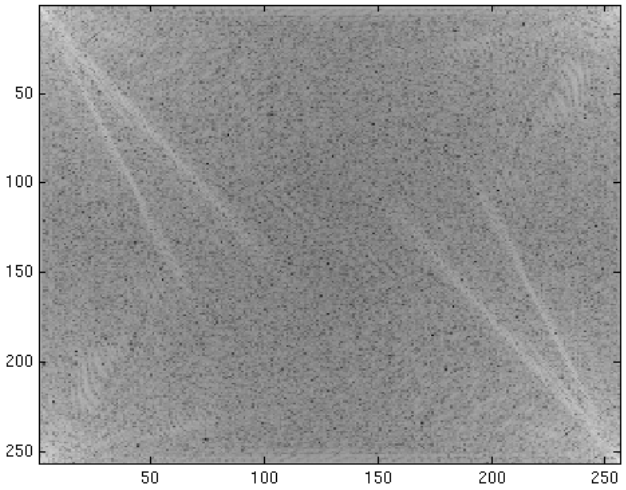
v= 0 1 2 3 4 …. N-5 N-4 N-3 N-2 N-1

u= 0

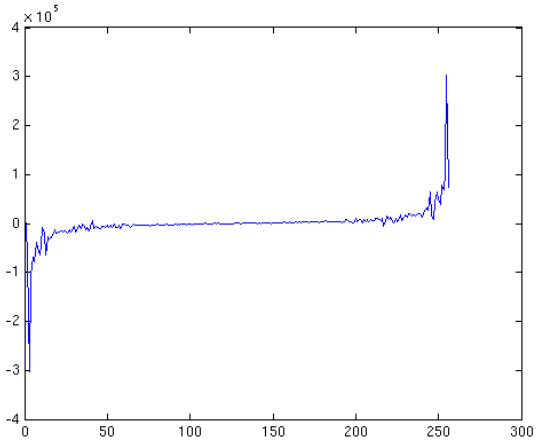
(antisymmetri i sinus-bildene)

Eksempel (symmetri)

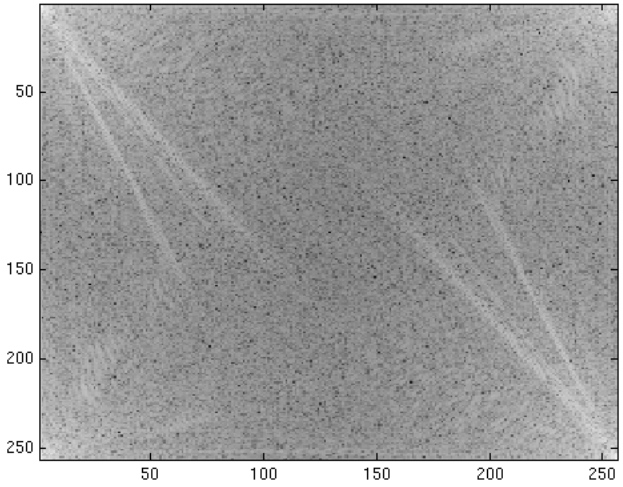




Logaritmen til absoluttverdien til cosinus-bidragene



Første linjen i sinus-bidragene



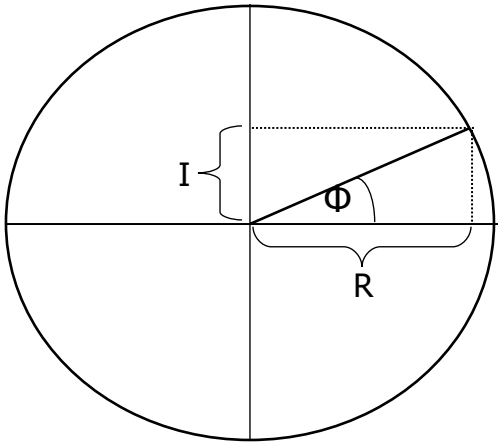
Logaritmen til absoluttverdien til sinus-bidragene

Finne fase og amplitude

- La R inneholde cosinus-bidragene og I inneholde sinus-bidragene.

- Amplituden til sinfunksjonen med frekvens u,v:

$$A = \sqrt{R(u,v)^2 + I(u,v)^2}$$



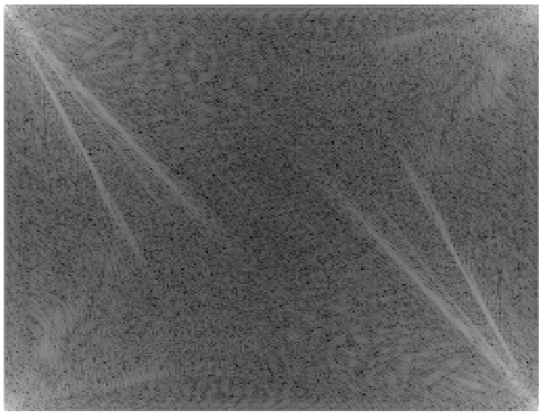
- Fasen til sinfunksjonen med frekvens u,v:

$$\phi = \tan^{-1}\left(\frac{I(u,v)}{R(u,v)}\right)$$

Husk fra s. 7: (A, Φ, Δ) <=> (A_1, A_2,Δ)

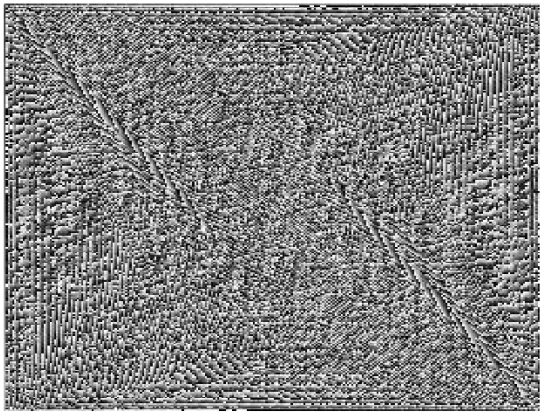
Eksempel: Amplitude og fase





(Log av) amplituden eller spekteret

Forteller noe om hvilke frekvenser bildet inneholder



$\phi(u,v)$ - fasen

Visuelt ser fasebildet ut som støy, men fasen inneholder viktig informasjon

Resultat som komplekst tall

- Letter håndtering (og utvider bruksområdet) ved å representere resultatet med *komplekse tall*: cosinus-bidragene i realdelen og sinus-bidragene i imaginærdelen

- La F beskrive bildet i den nye basisen

- F(u,v) = R(u,v) + jI(u,v) , $j = \sqrt{-1}$

- Amplitude og fase kommer da ut som modulus og argument (lengde og vinkel i komplekse planet), jfr *np.abs()* og *np.angle()*

2D diskret Fouriertransform (DFT)

$$F(u,v) = \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x,y)\, e^{-2j\pi(ux/N+vy/M)}$$

Husk at $e^{j\theta} = \cos(\theta) + j\sin(\theta)$, slik at vi ender opp med sin/cos-basisen vi er vant med:

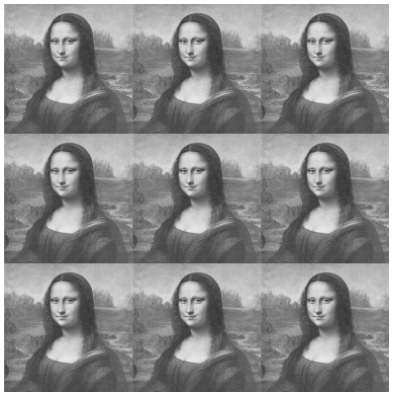
$$F(u,v) = \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x,y)\Big[\cos\big(2\pi(ux/N+vy/M)\big) + j\sin\big(-2\pi(ux/N+vy/M)\big)\Big]$$

Den inverse transformen:

$$f(x,y) = \frac{1}{NM} \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} F(u,v)\, e^{2j\pi(ux/N+vy/M)}$$

Egenskaper ved 2D DFT

- F(u,v) er periodisk:
  $F(u,v) = F(u+N,v) = F(u,v+N) = F(u+N,v+N)$



- Skal inverstranformen holde, må vi anta at bildet er periodisk:
  $f(x,y) = f(x+N,y) = f(x,y+N) = f(x+N,y+N)$

- Konjugert symmetri:
  Hvis f(x,y) er reell,
  er F(u,v) = conj(F(-u,-v))
  Altså er |F(u,v)|=|F(-u,-v)|

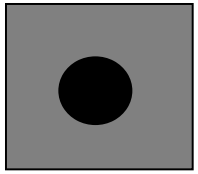
Om ikke annet er oppgitt, antar vi at N=M for enklere notasjon

Egenskaper ved 2D DFT, forts

- F(0,0) er proporsjonal med middelverdien i bildet
- Shift-teoremet: $f(x-x_0,y-y_0) \Leftrightarrow F(u,v)\ e^{-j2\pi(ux0+vy0)/N}$
- 2D DFT er separabelt i to 1D DFT
  - Absolutt nødvendig (sammen med FFT-algoritmen) for å beregningsmessig kunne transformere bilder av en viss størrelse

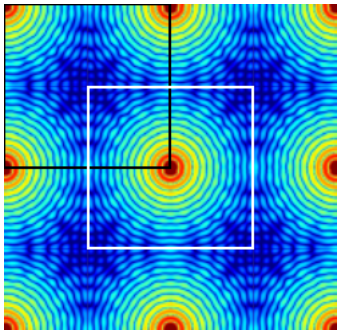
Framvisning av amplitudespekteret I/II

- Siden F(u,v) er periodisk med periode N, er det vanlig å forskyve spekteret slik at origo (u=v=0) ligger midt i bildet
  - Bytte kvadranter
  - [eller pre-multiplisere f(x,y) med $(-1)^{x+y}$]



f(x,y)

f(x,y): bildedomenet



|F(u,v)|

F(u,v): frekvensdomenet

|F(u,v)| kalles spekteret til f(x,y) (amplitudespekteret)

Retningen på bølgene følger retningen fra bildets sentrum

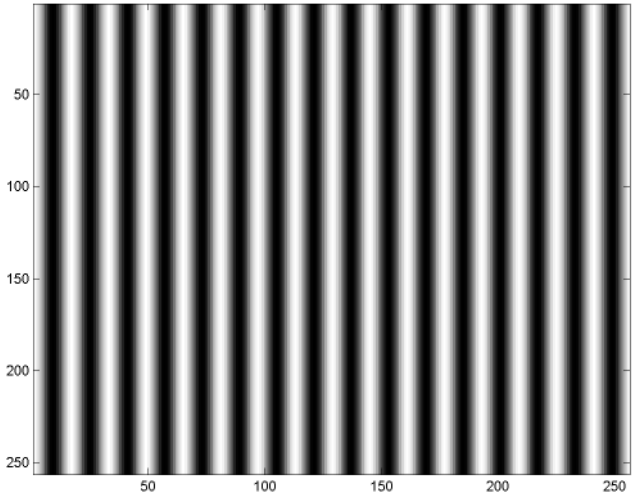
Framvisning av amplitudespekteret II/II

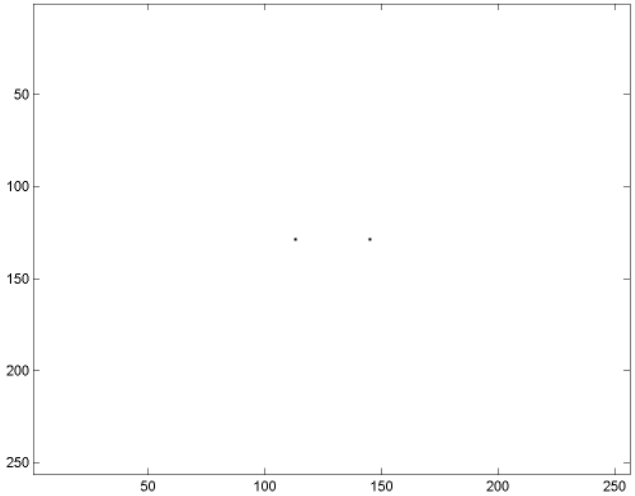
Skalering av verdier:

- Ofte stor dynamikk i |F(u,v)| (kan ha høye verdier)

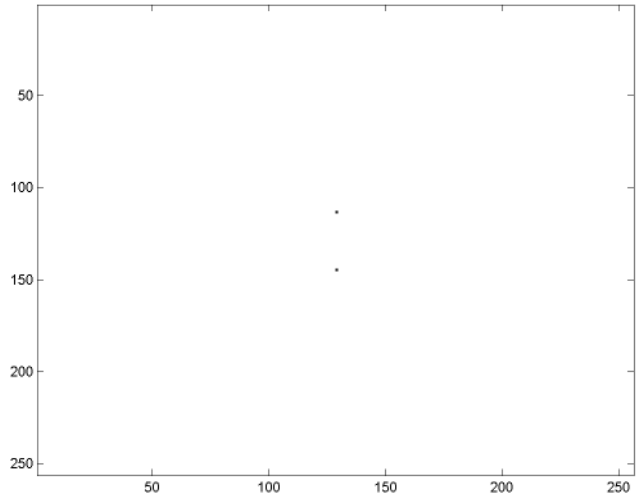
- Vanlig å benytte logaritmisk skala
  - Vi er mer interessert i **størrelsesforhold** i amplitude mellom frekvenser enn absolutt forskjell
  - g(u,v) = log(|F(u,v)|), som så eventuelt skaleres/affin-transformeres slik at man får gråtoner mellom 0 og 255 (8 bit)

Eksempler







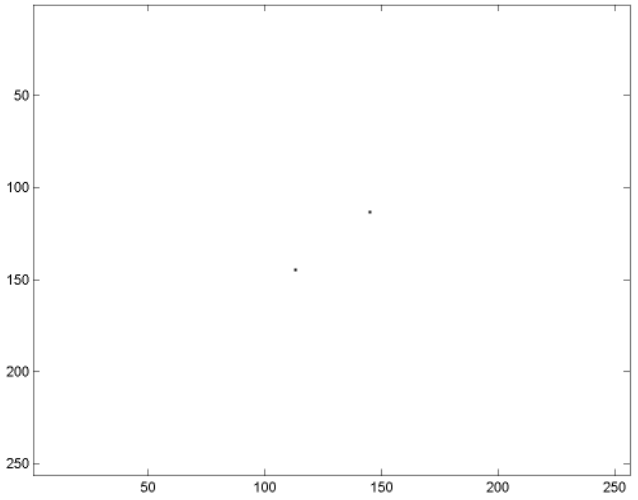


<- spekter

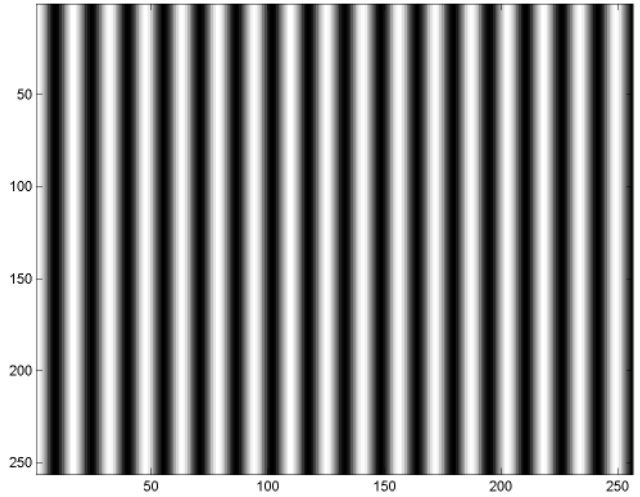
Her:
- Hvitt=0
- Nullfrekvensen midt i bildet

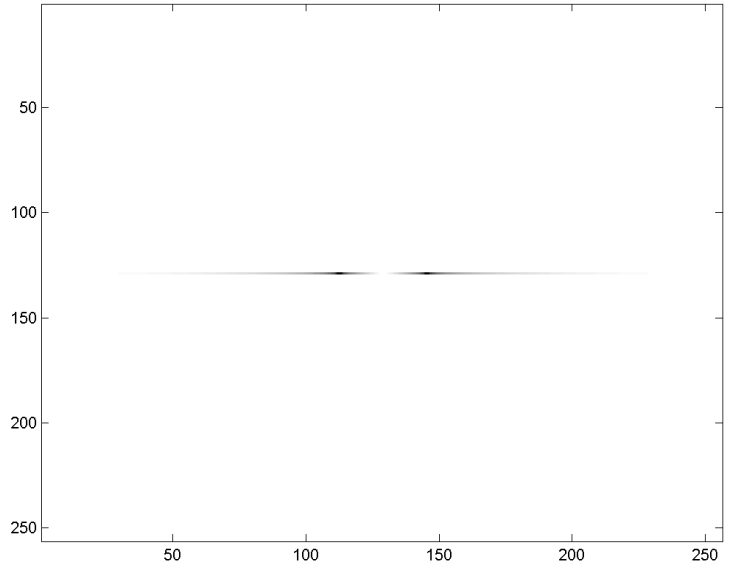
Eksempel – ”skrå” frekvens





Eksempel - diskontinuitet



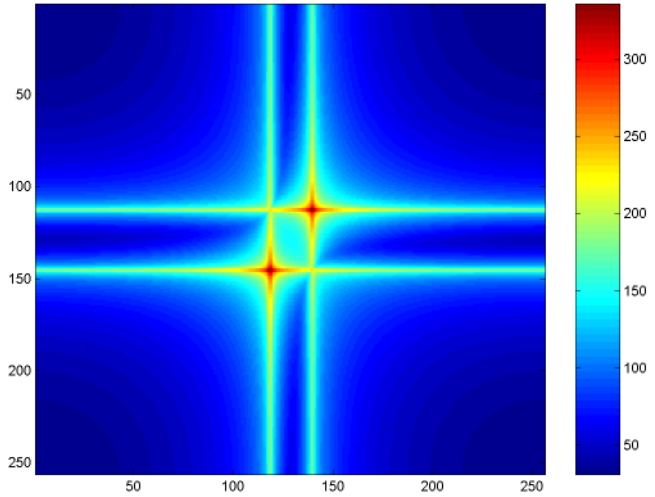


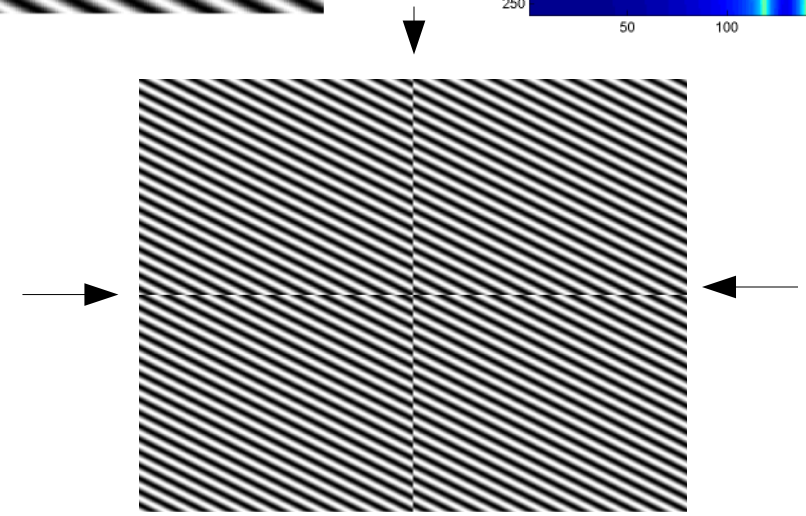
Ved å repetere bildet, ser vi tydelig kanter:



Eksempel – diskontinuitet II







Eksempel - vanlige objektformer

Examples of the Fourier transform for other simple shapes are shown be

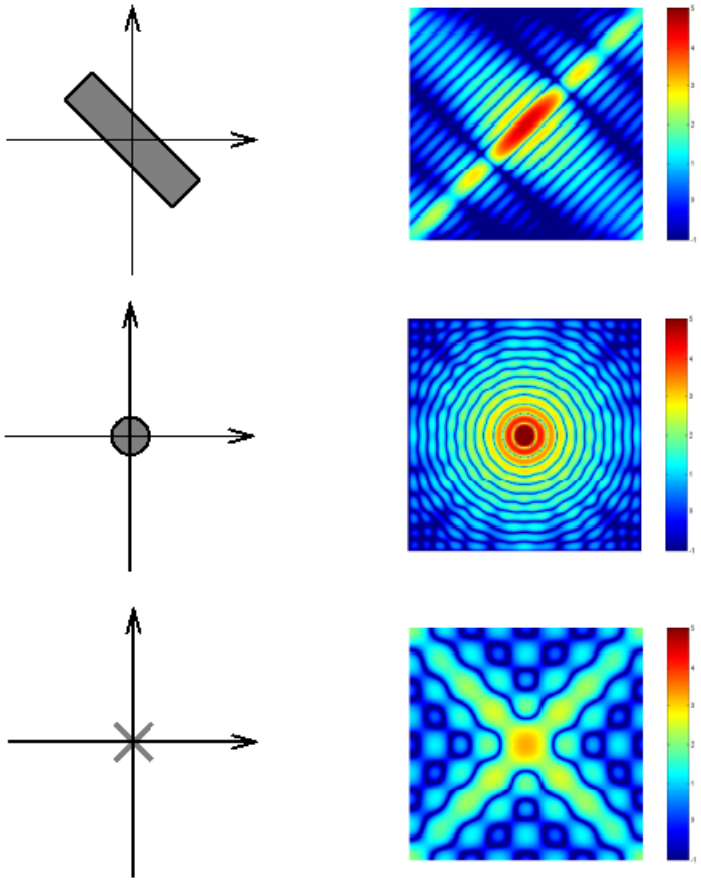
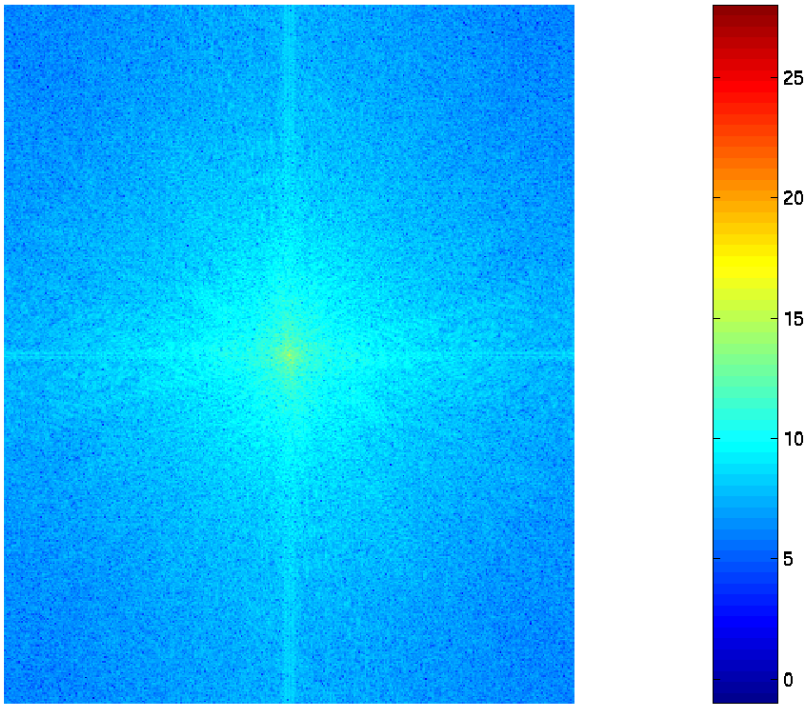


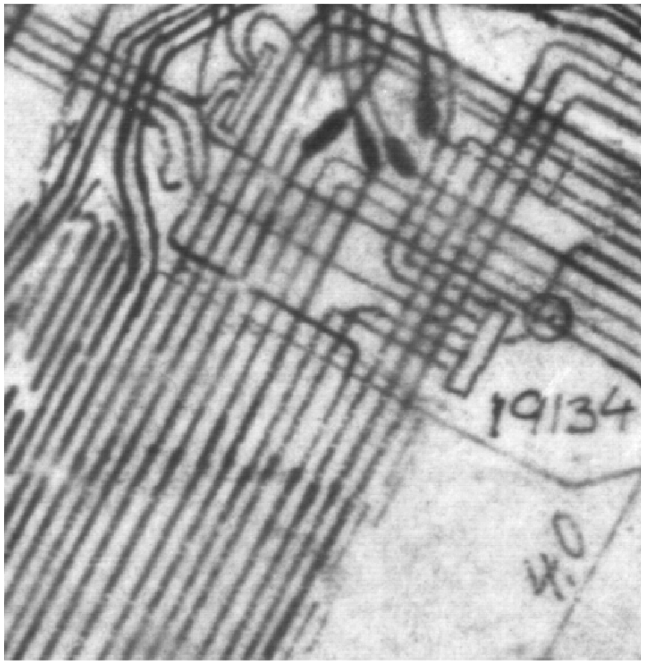
Figure 7-4: Fourier Transforms of Some Simple Shapes

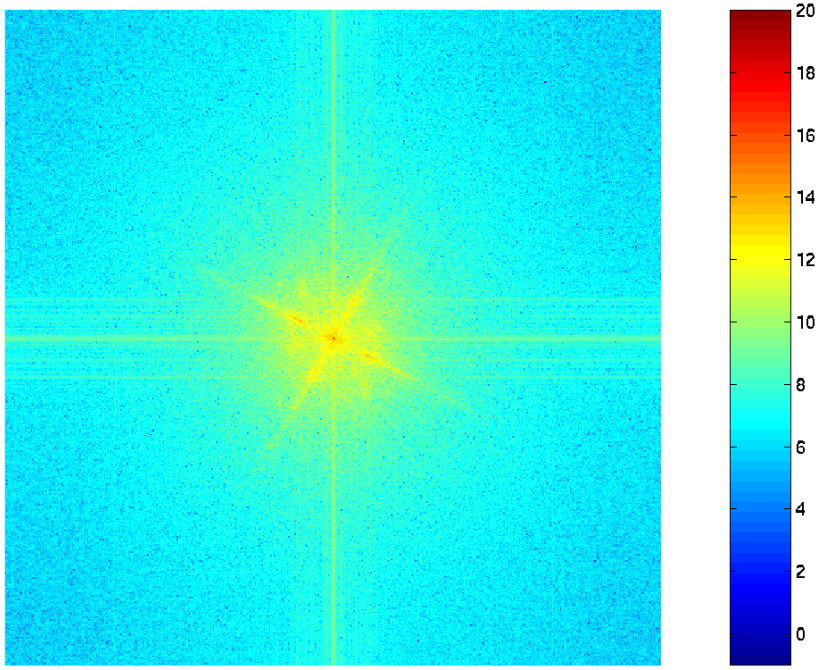
Eksempel – ”vanlig” bilde





Eksempel - retningsdominant

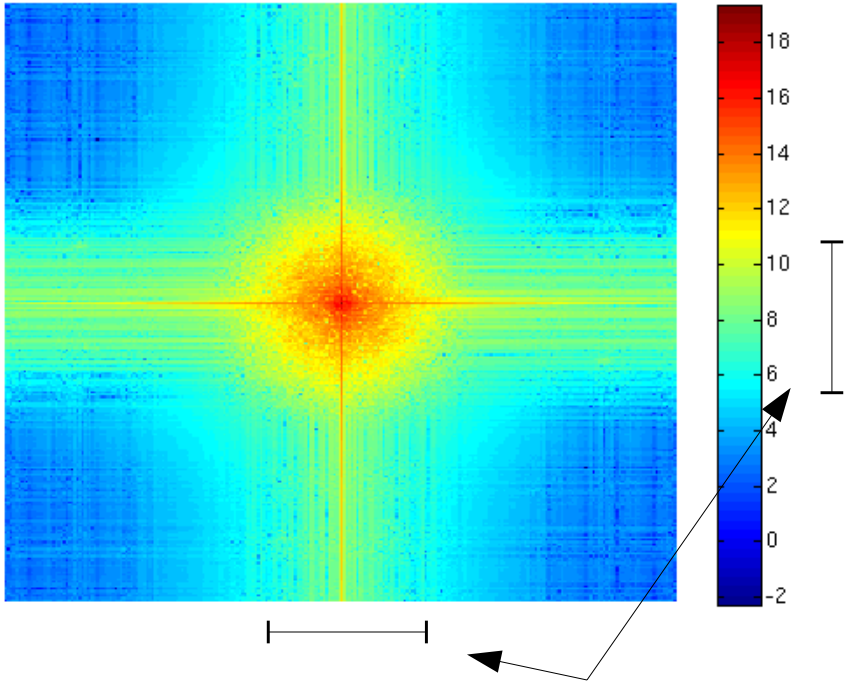




Eksempel - smal båndbredde



Lav oppløsning, lite detaljer



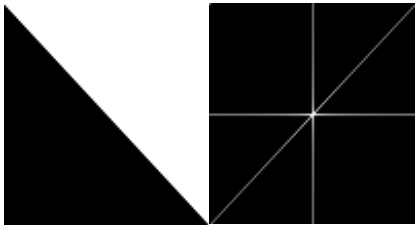
Så å si all «energien» er i dette smale området/båndet (både vertikalt og horisontalt)

Noen observasjoner ..

- Vanligvis størst bidrag/mest energi i spekteret for lave verdier av u,v
- Bidrag langs u- og v-aksen fordi bildet er implisitt periodisk og vi har diskontinuiteter langs kantene
- Linjestrukturer i gitt retning i bildedomenet har linjestruktur normalt på retningen i Fourier-domenet

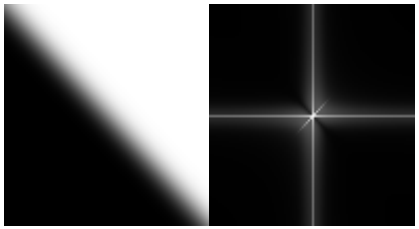
.. og noen observasjoner til

- Skarp kant:
  - Tilsvarer sum av mange sinusfunksjoner
  - Mange Fourier-koeffisienter er ≠ 0
  - Bredt bånd i Fourier-domenet



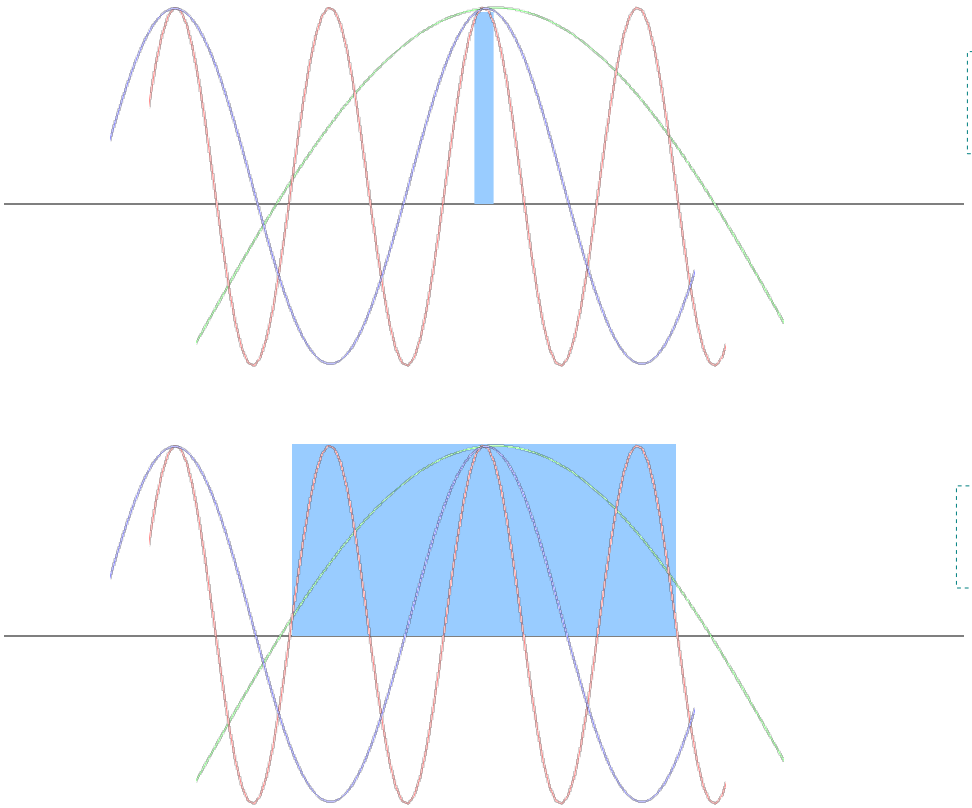
Bilde Fourier-spekter

- ”Blurret” kant:
  - Tilsvarer færre sinuser
  - Smalere bånd i Fourier-domenet



- Tommelfingerregler:
  - Smal struktur i bildedomenet : Bred struktur i Fourier-domenet
  - Bred struktur i bildedomenet: Smal struktur i Fourier-domenet
  - Linjestruktur i retning θ i bildedomenet: Linjestruktur i retning $\theta \pm 90°$ (normalt på) i Fourier-domenet

Intuisjonsbygging rundt smal struktur i bildedomenet -> bred struktur i Fourier-domenet, og omvendt



Høyt utslag/indreprodukt på **alle tre frekvensene**

Høyt utslag/indreprodukt **kun på laveste frekvens**

Implementasjon av DFT

- Beregning av F(u,v) for én u,v: $O(N^2)$
- Beregning for hele bildet: $N \times N$ F(u,v): $O(N^4)$
- Finnes en algoritme for rask beregning, 2D FFT (Fast Fouriertransform)
  - Benytter at Fourier-transformen er separabel i to 1D transformer
  - Bruker bilder (eller delbilder) med størrelse 2^k (k er heltall)
  - Har orden $O(N^2 \log_2 N)$

Fourier-transform i Python

```
F = np.fft.fft2(f)   # Gjør en 2D DFT-transform
f = np.fft.ifft2(F)  # .. og den inverse transformen (obs: mulige spor av imaginære avrundingsrester!)

F_r = np.real(F)     # Realdelen, altså cosinus-basis-bidragene
F_i = np.imag(F)     # Imaginærdelen, altså sinus-basis-bidragene

F_s = np.abs(F)      # Fourier-spekteret
F_p = np.angle(F)    # Fasen

F_r[u,v]             # Gir ”cosinus-bidragene” for frekvens u,v

F_r[0, 0]            # Gir ”DC-komponenten” / nullfrekvensen

np.fft.fftshift og np.fft.ifftshift: Flytter kvadranter slik at nullfrekvensen er i midten av bildet, samt omvendt

amp_log_shifted = np.fft.fftshift(np.log10(abs(F)))
plt.imshow(amp_log_shifted)
plt.imshow(amp_log_shifted, vmin=0)  # Her: Viser alle verdier <0 som sort
```

(Fourier-transform i Matlab/Octave)

- F = fft2(f); % Gjør en 2D DFT-transform
- f = ifft2(F); % .. og den inverse transformen (obs: mulige spor av imaginære avrundingsrester)

- F_r = real(F); % Realdelen, altså cosinus-basis-bidragene
- F_i = imag(F); % Imaginærdelen, altså sinus-basis-bidragene

- F_s = abs(F); % Fourier-spekteret
- F_p = angle(F); % Fasen

- F_r(u+1,v+1); % Gir ”cosinus-bidragene” for frekvens u,v

- F_r(1,1); % Gir ”DC-komponenten” / nullfrekvensen

- fftshift og ifftshift: Flytter kvadranter slik at nullfrekvensen er i midten av bildet, samt omvendt

- imagesc(fftshift(log(F_s)), [0 max(log(F_s(:)))]); % Her: Viser alle verdier <0 som sort

Oppsummering

- Sinus-funksjoner
  - 1D og 2D
  - «diskret» parametrisering: $\sin(\frac{2\pi(ux+vy)}{N})$
  - dekomponere *Asin(θi+Φ)* i sin- og cos-komponent

- Diskret Fourier-transform
  - bildet beskrevet med cos/sin-basisbilder (heltallige u, v)
  - indreprodukter
  - kompleks representasjon
    - cos- og sin-ledd som reell- og imaginær-komponent
  - fremvisning av spekteret |F(u,v)|
  - implisitt periodisitet
    - utslag i diskontinuitet -> ”ekstra” frekvenser
  - tommelfingerregler