TMA4123 - Kræsjkurs i Matlab.
Oppgavesett 2/3

28.02.2013

Oppgave 0: Bruk av `fftshift` og `ifftshift`

Når du bruker `fft` i Matlab flyttes frekvensene over midten av spekteret, slik at får du ut frekvensdata ordnet på en "unaturlig" måte. Den første komponenten svarer til frekvens 0, deretter kommer de positive frekvensene, og til slutt de negative frekvensene.

Dersom du ønsker å visualisere fouriertransformasjonen kan bruke `fftshift` til å "re-sentrere", slik at frekvens 0-komponenten havner i midten. Funksjonen `ifftshift` reverserer dette.

(a) Lag vektoren $v = [5, 2, 7, 3, 12]$ og beregn `fft(v)`. Sentrer deretter `fft(v)` ved å bruke `fftshift`. Legg merke til symmetrien om midten.

(b) Beregn `ifftshift(fftshift(v))` og sammenlign resultatet med v.

Hint: Skriv `doc fftshift` og `doc ifftshift` for beskrivelser av `fftshift` og `ifftshift`.

Oppgave 1:

Beregn den fouriertransformerte av $\sin(5t)$ over intervallet $[-\pi, \pi]$. Hvilke frekvenskomponenter skiller seg ut?

(a) Lag domenet $[-\pi, \pi]$, samplet i $n = 128$ punkter. Husk de periodiske randbetingelsene (første og siste fourierkomponent er identiske).

(b) Bruk `fft` til å plotte den fouriertransformerte slik at x-aksen i figuren er frekvensene/bølgenumrene.

Hint: Bruk fremgangsmåten gjennomgått i forelesningen (`enkel_fft_sech.m`).

Oppgave 2: Den fouriertransformerte til en gauss-funksjon

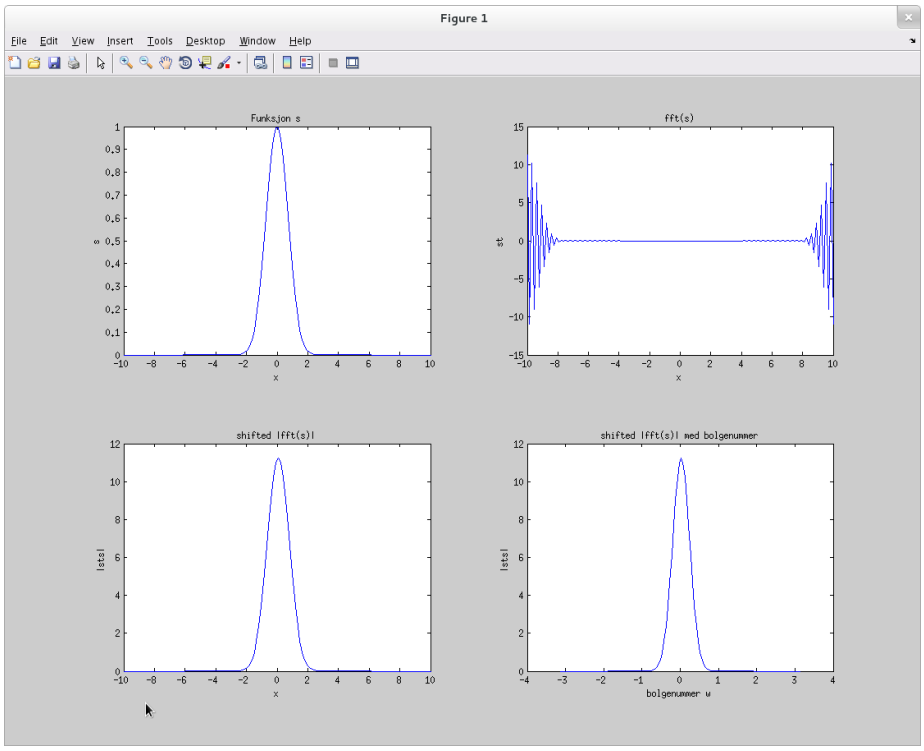
I denne oppgaven skal vi beregne den fouriertransformerte av funksjonen

$$s = e^{-x^2}$$

over intervallet $[-10, 10]$ og deretter rekonstruere funksjonen ved å bruke den inverse fouriertransformasjonen (`ifft`).

(a) Lag domenet $[-10, 10]$, samplet i $n = 128$ punkter. Husk de periodiske randbetingelsene.

(b) Bruk `fft` og `fftshift` til å gjenskape følgende figur. Legg merke til x-aksene i de to siste grafene.



(c) Rekonstruer funksjonen s ved hjelp av `ifft` og plott resultatet.

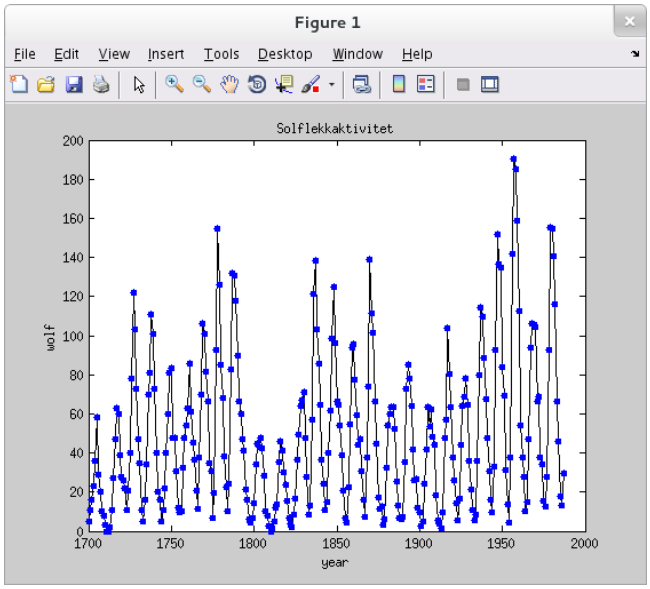
Hint: Følg prosedyren brukt i eksempelet gjennomgått i forelesningen.

Oppgave 3:

Solflekkaktiviteten på solen går i sykler på omtrent 11 år: hvert 11. år når den et maksimum. Se `http://en.wikipedia.org/wiki/Solar_cycle`.

I denne oppgaven skal vi bruke fouriertransformasjonen til å verifisere dette.[1]

(a) Astronomer har observert og loggført solflekkaktiviteten over lang tid. Vi skal bruke data samlet inn siden år 1700. Disse dataene er tilgjengelige i Matlab: skriv `load sunspot.dat` for å laste inn. Matrisen `sunspot` vil da inneholde disse dataene, med årstall i første kolonne. Den andre kolonnen inneholder de såkalte *Wolf-tallene* (`http://en.wikipedia.org/wiki/Wolf_number`) som beskriver størrelsen på solflekkaktiviteten. Lag et plot av årstall versus Wolf-tall.

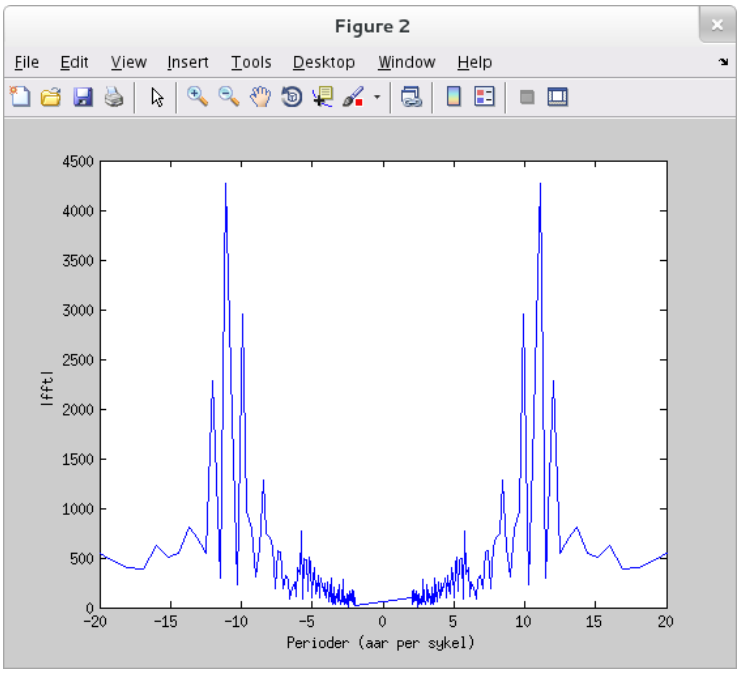


[1] Oppgaven er basert på `http://www.mathworks.se/products/matlab/examples.html?file=/products/demos/shipping/matlab/sunspots.html`

(b) Dersom størrelsen på Wolf-tallene går i sykler på ca. 11 år bør vi kunne detektere dette i frekvensdomenet. Finn fouriertransformasjonen `wolft` av Wolf-tallene.

(c) Vi ønsker å plotte dette, med syklene på x-aksen og absoluttverdien til `wolft` på y-aksen. For å gjøre resultatet lettere å tolke, dropp frekvens 0-komponenten fra den fouriertransformerte[2] ved å skrive `wolft(1)=[]`. Plot resultatet med
`plot(abs(fftshift(wolft)))`.

(d) Vi må fikse x-aksen slik at den viser *periodene*. Frekvensene finner vi i `(1/n)*[1:(n/2-1), -(n/2):-1]`, og plotter vi `wolft` mot dette ser vi antall sykler per år. Vi er interessert i periodene, som er gitt som 1/frekvensene. Forsøk å gjenskape følgende plot:



(e) Finn perioden som gir størst frekvenskomponent ved hjelp av `max`.

Hint: `[max,index] = max(signal)`.

[2] Forsøk å la være å droppe 0-komponenten for å se hvorfor. Denne frekvenskomponenten er veldig stor, noe som drukner den interessante delen.

Oppgave 4: Derivasjon ved hjelp av FFT

Vi kan bruke `fft` til å derivere funksjoner. Metoden[3] er basert på at

$$\mathcal{F}[f^{(n)}] = (iw)^n \mathcal{F}[f]. \tag{1}$$

Funksjonen vi ønsker å derivere er $f(x) = \cos x \sin 3x$, over området $[-2\pi, 2\pi]$.

(a) Definer området, samplet i $n = 128$ punkter. Definer også frekvensområdet k. Husk periodisitet (transformasjonen er lik i første og siste punkt).

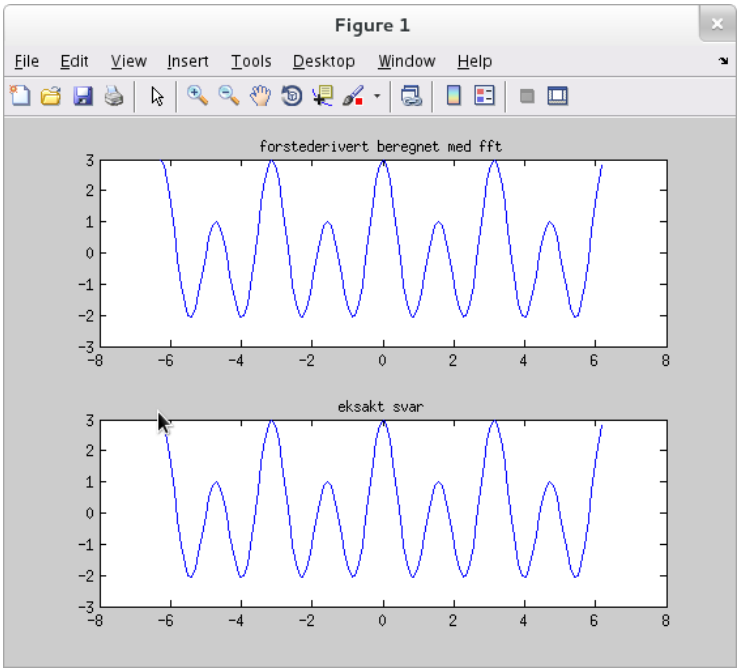
(b) Beregn den fouriertransformerte til f.

(c) Bruk formel 1 til å beregne de første- og andrederiverte til f. Pass på å multiplisere $\hat{f}$ med de korrekte verdiene av w.

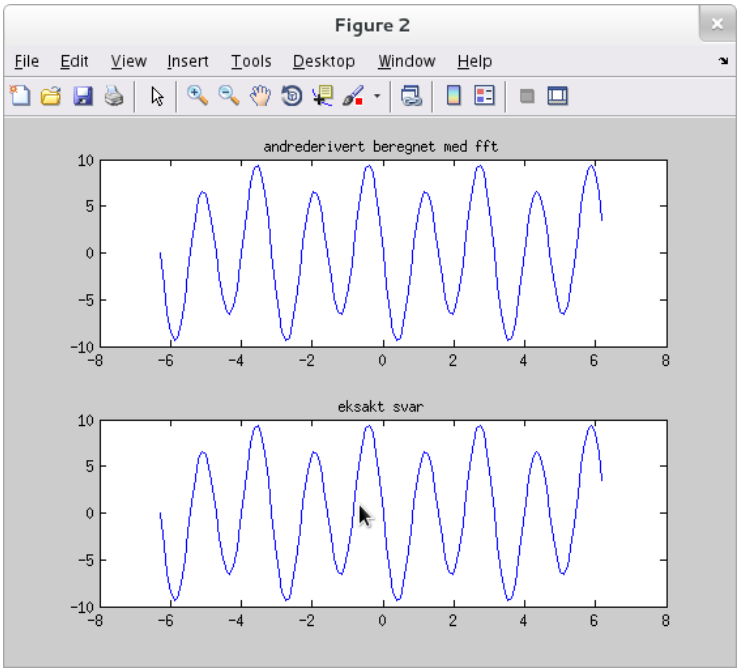
(d) Sammenlign med de eksakte svarene:

$$f'(x) = \cos(2x) + 2\cos(4x)$$
$$f''(x) = -2\sin(2x) - 8\sin(4x).$$

ved å konstruere følgende figurer:



[3]metoden er en såkalt "spektralmetode" (spectral method), og fungerer veldig bra på periodiske signaler/funksjoner



Oppgave 5: Forvrenging av lydsignal

I denne oppgaven skal vi forvrenge et lydsignal ved å først transformere det til frekvensdomenet, flytte litt rundt på frekvensene, og så transformere tilbake. Metoden kan brukes til å sende "hemmelige" meldinger over en åpen kanal. Dersom en kjenner hvordan frekvensene ble flyttet på er det en smal sak å rekonstruere signalet.

(a) Bruk `wavread` til å lese inn filen "chamberlain.wav" (lastes ned fra kursnettsiden). Kall signalet `sigin`. Signalet er samplet i $n = 201484$ punkter. Beregn lengden på lydfilen i sekunder.

Hint: `doc wavread`. Husk å lagre samplingsfrekvensen.

(b) Finn den fouriertransformerte `sigint` til lydsignalet. Plot resultatet ved hjelp av `plot` og `abs` (du trenger ikke bry deg om x-aksen og du behøver ikke bruke `fftshift`).

(c) Del opp `sigint` i fire like store deler (frekvensbånd), kalt A,B,C og D. (A vil inneholde samples fra 1 til n/4). Lag deretter et signal `sigoutt` ved å sette sammen delene i rekkefølgen C, D, B, A, og bruk den inverse fft til å danne signalet `sigout`. Du kan spille av dette ved hjelp av `sound(real(sigout),fs)`, der `fs` er samplingsfrekvensen til "chamberlain.wav".

(d) Last inn filen "scrambled.wav". Dette er dannet ved å gjennomføre forvrengingsmetoden skissert ovenfor. Rekonstruer signalet og lytt til resultatet. Det vanker en premie til førstemann som spiller av det rekonstruerte signalet i løpet av forelesningen.

(e) **Frivillig:** Forsøk å gjøre det forvrengte signalet fra (c) enda mer ugjenkjennelig, for eksempel ved å også snu om på rekkefølgene i frekvensene i noen av A,B,C,D (bruk `fliplr`) eller ved å dele opp i enda flere frekvensbånd.

(f) **Frivillig:** Last inn og forvreng en selvvalgt lydfil.

(g) **Frivillig:** Lag en matlabfunksjon (`scramble`) som tar en wav-fil som input og returnerer en forvrengt versjon. Funksjonen kan gjerne ta antall frekvensbånd som input. Lag også en funksjon som omgjør forvrengningen (`descramble`). Du vil ha nytte av `wavwrite`.