Høgskulen på Vestlandet	Avdeling for Ingeniørutdanning Institutt for Maskin- og Marinfag	**Øving 6**

MATLAB – Numerisk integrasjon, validering generelle integraler – `interp1`

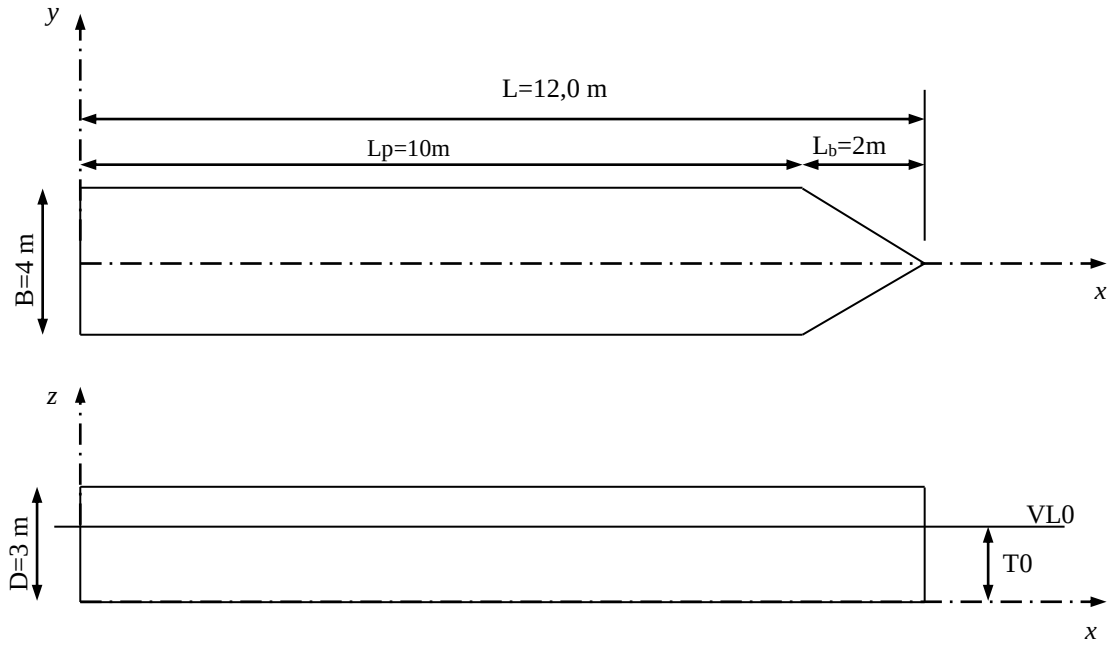
I forrige oppgave beregnet vi vannlinjeareal og videre egenskaper for kurvete geometrier. Nå skal vi bruke den generelle integralmetoden på et enkelt skrog for å validere våre integraler samt lære å interpolere, dvs sette inn punkter mellom gitte x og y verdier for å skape skritt med jevne mellomrom, s, i x-retning. For integrering med Trapez eller Simpson metoden trengs jo jevne skritt. Vi generaliserer derved hydrostatikkberegningen av et skrog ved å bruke numeriske integralmetoder istedenfor å regne med de eksakte formlene for firkanter og triangler. En numerisk metode er alltid en tilnærming, dvs det blir ikke lik løsning som den eksakte løsningen, som vi vil se i denne oppgave. Når resultatene blir lik på [mm] nivå i svar for BM eller GM, så er den numeriske metoden mer enn nøyaktig nok for våre beregninger. Skrogets størrelse og sundt fornuft avgjør hva som er nøyaktig nok.

Oppgave

a) Beregne tverrskips og langskips GM ved lettskip kondisjon for geometrien gitt i skissen under, bruk gitte dimensjoner. Lettskipstyngdepunktet er $KG_0 = D/3$. Legemet flyter uten trim og krengning ved dypgang $T_0 = 2.5$ m i en elv (ρ=1000kg/m^3). Begynn med symbolsk eksakt løsning ved å bruke formler for firkant og triangler som i de tidige øvingene 1-4.
(SVAR: I_{x0}=56m^4, I_{y0}= 450.99m^4, GM_{T0}=0.759m, GM_{L0}=4.3499m. Løsningsforslag på Canvas.)

b) Gjør oppgaven ved å bruke Trapez-integrasjon. Observer at det kan være en god idé at dele in skrittene sånn at hjørnet ved x = 10m blir diskretisert (inkludert). Forfine skrittet til GMT overensstemmer med den eksakte verdien på 3:e desimalen, [mm] nøyaktighet.
Se MATLAB anvisning på side 2. MATLAB løsningen på Canvas viser både a og b løsning.

c) Erstatt beregningene for vannlinjearealet (Aw, xF, Ix og Iy) med en egendefinert funksjon, sånn at disse ligningene enkelt kan gjenbrukes for andre vannlinjeplan, se anvisninger for MATLAB på side 4.



Anvisninger MATLAB

Begynn med å lagre et nytt skript med navn Ov6.m og husk å skrive `clear all` i begynnelsen av skriptet for å slette gamle variabler som kan ligge i din *Workspace*. Definer alle gitte verdier:

```
% Geometri fra oppgevebeskrivelsen og ferskvann:
rho=1000; % [kg/m^3] tetthet ferskvann
L=12;     % [m] Lengde
Lb=2;     % [m] Lengde baug
Lp=L-Lb; % [m] Lengde paralellskip
B=4;      % [m] Bredde
D=3;      % [m] Dybde
T0=2.5;   % [m] Dypgang lettskip
KG0=D/3; % [m] Vertikal lettskips tyngdepunkt [x y z]-koordinat
```

Siden skipet har vertikale kanter og flyter ved null trim, så er vannlinjearealet likt for hver dypgang og man kan allerede konstatere at flotasjonssenter, oppdriftsenter og tyngdepunkt ligger på en vertikal linje, dvs de har samme horisontale *x* og *y* koordinater. Man kan også konstatere at man ikke trenger å integrere volumdeplasementet ved hjelp av vannlinjeplan eller tverrskipsplan pga de vertikale kantene. Hvis det ikke hadde vært sånn, må man først beregne vannlinjearealer som funksjon av dypgang og sen integrere. Vannlinjeplan og tverrskipsplan brukes då også for å regne 1. ordens moment for volumer ved beregning av Oppdriftsenter (tyndepunktsatsen for volumer).

Med dette i bakgrunnen begynner vi oppgaven. Det eneste vi trenger å integrere er vannlinjeplan, 1.ordensmoment for flotasjonssenter og arealtreghetsmoment for vannlinjeplan. Lag to vektorer som beskriver skipets enkle geometri ved *x* og *y* verdier på vannlinjeplanet, dvs *y* er halvbredden og *x* er i lengderetning.

```
% Geometri for integrasjon, x-y koordinater:
y1 = [B/2 B/2  0]; % B/2 y locations, using symmetry
x1 = [0   Lp   L];
```

For å kunne integrere med trapez metoden trenger vi et jevnt skritt i x. Skap en x-vektor med jevnt skritt fra null til lengden på skipet. Bruk matlabfunksjonen `interp1` for å finne motsvarende y-verdier ved de nye x-verdiene. Begynn med et skritt på 1 meter.

```
x=[0:1:L];
y=interp1(x1,y1,x);
```

Tegne en figur sånn at du ser at geometrien stemmer og at du har fått en punkt i knekken ved x=Lp, dvs at diskretiseringen inkluderer denne punkt.

```
% Plot for å se formen
plot(x,y,'-o')
xlabel('x - Skip lengde')
ylabel('y - Skip halv bredde')
```

Kjør programmet, hvis du ikke allerede har gjort det (F5). Man ser at figuren ikke er i samme skala på x og y akse, Matlab plotter per automatikk sånn att figuren fyller opp i et kvadratisk vindu, alltid med samme størrelse. Derfor legg til etter ylabel:

```
axis equal
```

For å få litt luft ovenfor og til høyre i figuren, kan man sette eget intervall på akslene ved å legge til:

```
axis([0 L+1 0 B/2+1]) % axis([xmin xmax ymin ymax])
```

Se `>>doc axis` for å finne ut av flere ting man kan gjøre med akslene i en figur.

For den numeriske integrasjonen trengs integrasjonssteget s, som vi automatisk skriver som avstanden mellom andre og første verdien i den nye x-vektoren.

```
s = x(2)-x(1);  % integrasjonssteg i x-retning
```

Trapez metoden er innebygget i MATLAB, men for øving lagrer vi en Trapez faktor vektor på samme sett som for Simpson metoden. Unnvik å kalle denne for `tf`, siden det finnes en kommando i Matlab for `tf` (transfer function). Det er ingen fare ved å skrive tf men då skriver man over denne funksjon og den går ikke å bruke i sin kode. Eksempelvis, på samme sett som om man skulle definere innebyggd `pi` som noe annet en verdien på akkurat π. Så det er helt enkelt kod-etikk at ikke bruke slike innebygde funksjoner som variabelnavn når man vet om at det er en funksjon.
Man vet at trapezfaktor vektoren skal begynne med 1 og slutte med 1 og ha 2:er emellom. Automatiseres dette så skal antall 2:er være lik lengden på x-vektoren minus 2.

```
trf=[1 2*ones(1,length(x)-2) 1];
```

Nå, beregne vannlinjearealet med trapezmetoden:

```
Aw= s/2*(trf*y') *2
```

Eller via innebygget funksjon i Matlab:

```
Aw= trapz(x,y) *2
```

Beregne deplasementet:

```
nabla0=Aw*T0;  % gjelder bare ved vertikale kanter
delta0=nabla0*rho
```

Beregne flotasjonssenter, oppdriftssenter og tyngdepunkt:

```
My= s/2 * trf*(x.*y)' *2    % 1.ordens moment
```

Eller via innebygget funksjon i Matlab:

```
My= trapz(x, x.*y) *2

xF0= My/Aw
yF0=0; % pga symmetri

xB0=xF0;  % pga vertikale kanter, null trim
yB0=yF0;  % pga vertikale kanter, null krengning
zB0=T0/2; % pga vertikale kanter, ellers zB=(1. ordens moment)/nabla0

xG0=xB0;  % Pga null trim
yG0=yB0;  % pga null krengning
zG0=KG0;
```

Beregne BM langskips og tverrskips. I_x og I_y trengs, integrer 2. ordens moment og observer at I_y i første trinn blir rundt skipets globale koordinatsystem ved hekken:

```
Ix0=1/3*s/2*(trf*(y.^3)') *2
```

Eller via innebygget funksjon i Matlab:

```
Ix0=1/3*trapz(x,y.^3) *2 % Gjelder bare for symmetriske skip, ikke
                         %  katamaran
Iy0_hekk=s/2*(trf*(x.^2.*y)') *2
```

Eller via innebygget funksjon i Matlab:

```
Iy0_hekk=trapz(x, x.^2.*y) *2
```

Bruk omvendt Steiners sats for å regne ut I_y rundt flotasjonssenter (I_x, er jo allerede i flotasjonssenter pga symmetri).

```
Iy0=Iy0_hekk-Aw*(xF0)^2
```

BM og GM:

```
BMT0=Ix0/nabla0
BML0=Iy0/nabla0
```

```
GMT0=zB0+BMT0-zG0
GML0=zB0+BML0-zG0
```

Jamfør svaret med det eksakt beregnede i oppgave a). Gjør skrittet mindre i x-vektoren i begynnelsen av ditt skript for å få en bedre tilnærmet integrasjon sånn at en millimeter nøyaktighet i GM_T blir oppnådd, feks:

```
x=[0:0.1:L];
```

Man kan tenke seg at Trapez integrasjonen burde gi eksakt svar for dette eksempel når alle linjer er linær. Dette stemmer for A_w arealet men ikke for 1. ordens moment. Det man nemlig gjør med Trapez (og Simpson) integrasjon, er at man angir vektarmen til arealet i 1. ordens moment, til venstre kant på arealet, og ikke til senter på arealet. Derfor blir tilnærmingen bedre for minsket skritt, når vektarmen blir mer og mer rett.

Oppgave c)

Lag ditt Ov6.m script som ett nytt, f.eks. **Ov6c.m**, før du begynner på oppgaven.
I tillegg: Åpne et nytt skript via **New Script** og spare dette som **waterplane_cals.m** i samme mappe som Ov6c.m skriptet. Du skal nå ha to fliker med ditt Ov6c.m og waterplane_cals.m skript i *Editorn*.

Vi skal nå bruke vårt Ov6c.m skript som hoved skript (Main script) og fra dette anrope sub-funksjonen waterplane_cals.m skriptet, hvor vi snart skal legge beregningene for vannlinjeplanet. Alle funksjoner i MATLAB er bygget opp på denne prinsipp (så som sinus, interp1,mm). Nå skal vi altså bare skape en egen funksjon som vi vil tilpasse for våre beregninger.

waterplane_cals.m vil altså bli brukt som en funksjon med input og output. Husk, f.eks. for `interp1` hade vi (`x1,y1,x`) som input og `y` var det output som funksjonen beregnet for oss. Nå skal vi gi x og y koordinatene for vannlinjeplanet som input til en funksjon som skal regne ut vannlinjeareal, flotasjonssenter og arealtreghetsmoment. Dette blir vår funksjons output.

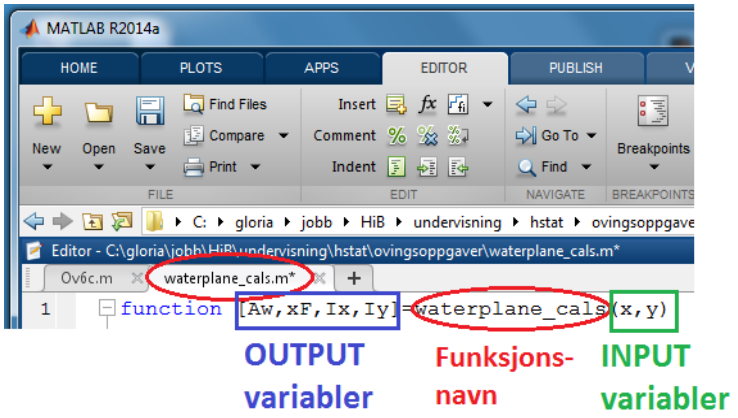
Gå til waterplane_cals.m skriptet som nå er blankt. Begynn med å skrive:

```
function
```

Du vil se at teksten automatisk blir blå. Etter denne på samme rad legg til:

```
function [Aw,xF,Ix,Iy]=waterplane_cals(x,y)
```

Her er det som står til venstre i firkantparentes output fra funksjonen og det som står i vanlig parentes bak funksjonsnavnet er input til funksjonen. Posisjonene for hver variabel i disse parenteser, så kalte *argument*, er viktige å holde kontroll på og de er separert med komma tegn. Det er også viktig at funksjonsnavnet **waterplane_cals** er det samme som navnet på m-filen.



Kopier teksten etter function og gå tilbake til hovedskriptet Ov6c.m. Etter linjen der du har gjort jevne skritt med interp1, dvs hvor ferdige x og y allerede er definert for å kunne integreres, lim inn det du har kopiert:

```
[Aw,xF,Ix,Iy]=waterplane_cals(x,y)
```

Posisjonene for *argumenten* inneholder informasjonen som kommer ut fra funksjonen og ikke variabelnavnet, man kan derfor døpe output til det man vil. Nå regner vi for eksempel på lettskipstilstand og seinere kanskje vi vil bruke samme funksjon når vi har vekt på skipet (med andre x og y verdier). Så døp om output til:

```
[Aw0,xF0,Ix0,Iy0]=waterplane_cals(x,y)
```

Gå nedover i ditt hovedskript og klipp ut radene der du har beregnet skrittet *s*, trapezfaktor vektor, Aw0, xF0, Ix0 og Iy0, dvs inklusive Steiners. Vi vil få ut Ix0 og Iy0 rundt flotasjonssenter direkte fra funksjonen.

Gå tilbake til waterplane_cals.m og klistre in dine ligninger en etter en. Tenk på at døpe variablene som skal være output likt som i første raden i funksjons-skriptet, dvs skriv Aw og ikke Aw0. Skriptet skal totalt se ca slik ut:

```
function [Aw,xF,Ix,Iy]=waterplane_cals(x,y)
s = x(2)-x(1);  % integration step
trf=[1 2*ones(1,length(x)-2) 1]; % trapez factor vector

Aw= s/2*trf*y' *2;           % [m^2] Water plane area
My= s/2 * trf*(x.*y)' *2 ;   % 1st order moment about y
xF= My/Aw;                   % [m] Center of Flotation (F)
Ix=1/3*s/2*(trf*(y.^3)') *2 ; % 2nd order moment about x-axis
Iy_hekk=s/2*(trf*(x.^2.*y)') *2;  % 2nd order moment about y-axis at
stern
Iy=Iy_hekk-Aw*(xF)^2 ;       % 2nd order moment about y axis at F.
```

Trykk på **Save** sånn at waterplane_cals.m spares i denne siste oppdaterte versjon og gå tilbake til Ov6c.m. Dobbelshekk at du utfører alle beregningen etter at du anroper funksjonen. Den skal alltså være plassert over deplasement beregningen der du bruker Aw0. Skriptet skal bare inneholde tyngdepunkter, oppdriftsenter, og sedan BM og GM ligningene samt at yF0=0. Se ferdig skript på It's Learning Ov6c.m.

Kjør Ov6c (F5) og du skal få samme svar som tidligere. Skriptet har nå blivit mer oversiktlig og du har plassert ligninger som du kan gjenbruke seinere i en funksjon.
Til sist, funksjoner er viktige å kommentere siden disse er vanlig å dele med seg for andre som vil bruke de. For denne funksjon er det viktig å si at den bare gjelder for symmetriske skip rundt x-aksen (vi beregner jo ikke flotasjonssenter i y retning og vi bruker halvbredde som input). Også, Ix beregningen vi gjør, gjelder bare for et areal, dvs ikke for en katamaran hvor to arealer er fordelt rundt x-aksen (då måtte vi brukt annen input en halvbredde, og Steiners sats rundt x-aksen også). Det er derfor like viktig å beskrive hvordan input skal se ut og hva output blir. Det pleier å se ca. slik ut at man legger til kommentaren direkte under function:

```
function [Aw,xF,Ix,Iy]=waterplane_cals(x,y)
% Calculates properties of a waterplane area, which is symmetric
% around the longitudinal x-axis of the ship. Only single area, no catamaran.
% Integration by Trapez method.
% INPUT:   x: ship length vector with equal step size in direction stern to bow
%          y: half breadth vector of waterplane along x
% OUTPUT: Aw: Waterplane area
%         xF: x distance to centre of flotation
%         Ix: 2nd moment of inertia around x-axis through flotation centre
%         Iy: 2nd moment of inertia around y-axis through flotation centre
% //Gloria Stenfelt, February 2015
```