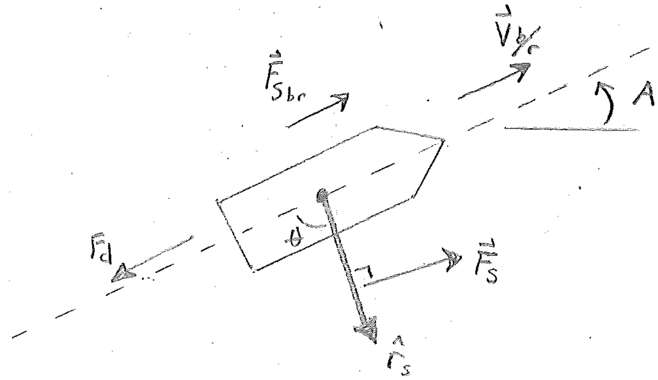
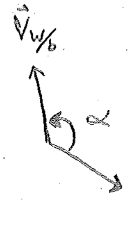
THIS IS THE VECTOR-EQUIVALENT OF BRYSON'S SAILBOAT MODEL





$\hat{r}_s \times \vec{V}_{w/b} = |\hat{r}_s||\vec{V}_{w/b}| \sin(\alpha)$

EQUATIONS OF MOTION:

KINEMATIC EOM ARE FOUND BY ASSUMING A ZERO-FORCE BALANCE OVER EACH TIME INTERVAL.

$$\Sigma \vec{F} = 0$$

$$= \vec{F}_{s_{br}} - \vec{F}_d$$

$\hat{V}_{b/r} = \cos(A)\hat{\imath} + \sin(A)\hat{\jmath}$

$$\rightarrow \vec{F}_{s_{br}} = |\vec{F}_{s_{br}}| \hat{V}_{b/r}$$

$$= |\vec{F}_s| \sin(\theta) \hat{V}_{b/r}$$

$$= |\vec{F}_s| (-\hat{V}_{b/r} \times \hat{r}_s) \hat{V}_{b/r}$$

$\vec{V} \times \vec{u} = \begin{vmatrix} V_x & V_y \\ u_x & u_y \end{vmatrix} = V_x u_y - V_y u_x$

$= \det\left(\begin{bmatrix} \vec{V}' \\ \vec{u}' \end{bmatrix}\right)$

$$\rightarrow |\vec{F}_s| = c_1 |\vec{V}_{w/b}|^2 \sin(\alpha)$$

$$= c_1 |\vec{V}_{w/b}| (\hat{r}_s \times \vec{V}_{w/b})$$

$$\vec{F}_{s_{br}} = c_1 |\vec{V}_{w/b}| (\hat{r}_s \times \vec{V}_{w/b}) (-\hat{V}_{b/r} \times \hat{r}_s) \hat{V}_{b/r} \quad \leftarrow$$

$$\rightarrow \vec{F}_d = c_2 |\vec{V}_{b/r}|^2 \hat{V}_{b/r}$$

$$= c_2 |\vec{V}_{b/r}| \vec{V}_{b/r} \quad \leftarrow$$

$\mu^2 = \frac{c_1}{c_2}$

$$\Sigma \vec{F} = 0 = \mu^2 \left[|\vec{V}_{w/b}| (\hat{r}_s \times \vec{V}_{w/b}) (-\hat{V}_{b/r} \times \hat{r}_s) \hat{V}_{b/r} \right] - |\vec{V}_{b/r}| \vec{V}_{b/r}$$

WE ARE LOOKING FOR THE ABSOLUTE VELOCITY OF THE BOAT. THIS IS FOUND BY SOLVING THE SRT:

$$\left\{\begin{array}{l} 0 = \mu^2\left[|\vec{V}_{w/b}|(\hat{r}_s \times \vec{V}_{w/b})(-\hat{V}_{b/r} \times \hat{r}_s)\hat{V}_{b/r}\right] - |\vec{V}_{b/r}|\vec{V}_{b/r} \\ \vec{V}_{w/b} = \vec{V}_w - \vec{V}_b \\ \vec{V}_{b/r} = \vec{V}_b - \vec{V}_r \end{array}\right\}$$

(i.e. $\vec{V}_b = \vec{f}(A, \theta, \vec{V}_r, \vec{V}_w)$)

A CLOSED-FORM SOLUTION DOES NOT EXIST, SO USE A SOLVER IN MATLAB. I FOUND fmincon() TO BE MORE ROBUST THAN fsolve().

---

a) FORMAL STATEMENT OF OPTIMIZATION PROBLEM

MINIMIZE: $J = t_f$

SUBJECT TO: $x_{i+1} = x_i + V_{x_i}\Delta$

$y_{i+1} = y_i + V_{y_i}\Delta$

$\vec{V} = \vec{V}_b \equiv$ ABSOLUTE VEL

CONSTRAINTS: $x_0 = 85m \qquad y_0 = 0$

$x_f = 115m \qquad y_f = 200m$

$0 \le y_i \le 200m \quad \forall\, t_i$

$\measuredangle(V_{b/r})_i = A_i \quad \forall\, t_i$

ANGLE OF VEL BOAT wrt RIVER IS THE SAME AS HEADING ANGLE (BOAT CAN'T GO BACKWARDS OR LATERAL)

↳ COMPARE TO BOLZA FORM:

$$\left\{\begin{array}{ll} \text{COST:} & J = \phi(\bar{x}(N),\Delta) + \sum_{i=0}^{N-1} L(\bar{x}(i), \bar{u}(i), \Delta) \\ \text{STATE EQNS:} & \bar{x}(i+1) = \bar{f}(\bar{x}(i), \bar{u}(i), \Delta) \qquad \bar{x}(0) = \bar{x}_0 \\ \text{TERM CONST:} & \bar{\psi}(\bar{x}(N),\Delta) = 0 \end{array}\right\}$$

THUS:

$$\phi(\bar{x}(N),\Delta) = N\Delta$$

THUS, BOLZA & MAYER ARE EQUIVALENT

$$L(\bar{x}(i), \bar{u}(i), \Delta) = 0$$

$$\bar{x} = \begin{bmatrix} x \\ y \end{bmatrix} \qquad \bar{f}(\bar{x}, \bar{u}, \Delta) = \begin{bmatrix} x + V_x\Delta \\ y + V_y\Delta \end{bmatrix}$$

$$\bar{\psi}(\bar{x}(N),\Delta) = \begin{bmatrix} x(N) - x_f \\ y(N) - y_f \end{bmatrix}$$

FIRST VARIATION OF AUGMENTED LAGRANGIAN

BRYSON

CONST.

$$\frac{d}{dx}\left(\lambda^T(0)\, x_0\right) = 0$$

$$\delta \bar{J} = \left[\phi_{\bar{x}}(\bar{x}(N),\Delta) + \bar{\nu}^T \bar{\psi}_{\bar{x}}(\bar{x}(N),\Delta) - \lambda^T(N)\right]\delta\bar{x}(N) \tag{4.9}$$

$$+ \left[\phi_\Delta(\bar{x}(N),\Delta) + \bar{\nu}^T \bar{\psi}_\Delta(\bar{x}(N),\Delta)\right]\delta\Delta$$

$$+ \sum_{i=0}^{N-1}\Big[\left(H_x(\bar{x}(i),\bar{u}(i),\Delta) - \bar{\lambda}^T(i)\right)\delta\bar{x}(i) + H_u(\bar{x}(i),\bar{u}(i),\Delta)\,\delta\bar{u}(i)$$

$$+ H_\Delta(\bar{x}(i),\bar{u}(i),\Delta)\,\delta\Delta\Big]$$

FIRST-ORDER OPTIMAL CONDITION REFERS TO A STATIONARY POINT, i.e. ONE THAT DOES NOT CHANGE W.r.t INPUTS (IN THIS CASE, INPUTS ARE $\bar{u}$ & Δ). CLEARLY, A STATIONARY POINT, $\bar{X}^*$, CHANGES WITH $\bar{X}$. THEREFORE, SET ALL COEFFICIENTS OF $\delta\bar{x}$ TO ZERO.

COSTATE

$$H_{\bar{x}}(i) - \bar{\lambda}^T(i) = 0 \tag{4.10}$$

$$\Rightarrow \bar{\lambda}^T(i) = \bar{\lambda}^T(i+1)\, f_{\bar{x}}(i)$$

$$\Phi_{\bar{x}}(N) - \bar{\lambda}^T(N) = 0 \tag{4.11}$$

$$\Rightarrow \bar{\lambda}^T(N) = \phi_{\bar{x}}(N) + \bar{\nu}^T\bar{\psi}_{\bar{x}}(N)$$

THUS, (4.9) BECOMES

$$\delta\bar{J} = \left[\Phi_\Delta(N) + \sum_{i=0}^{N-1} H_\Delta(i)\right]\delta\Delta + \sum_{i=0}^{N-1} H_u(i)\,\delta\bar{u}(i) \tag{4.12}$$

FOR A STATIONARY POINT, $\delta\bar{J} = 0$ FOR FINITE, NONZERO, AND ARBITRARY $\delta\Delta$ & $\delta\bar{u}$. THEREFORE, THE COEFFICIENTS MUST BE ZERO.

CONTROL

$$H_u(i) = 0 \quad \forall\ i = 0, \dots, N-1 \tag{4.13}$$

TRANSVERSALITY

$$\Phi_\Delta + \sum_{i=0}^{N-1} H_\Delta(i) = 0 \tag{4.14}$$

$$\phi_\Delta(\bar{x}(N),\Delta) + \bar{\nu}^T\bar{\psi}_\Delta(\bar{x}(N),\Delta) + \sum_{i=0}^{N-1} H_\Delta(i) = 0$$

FOR THIS PROBLEM, THE FIRST-ORDER VARIATION CRITERIA ARE

STATE EQUATIONS $\bar{x}_{i+1} = \bar{f}(\bar{x}(i), \bar{u}(i), \Delta)$

$$x_{i+1} = x_i + V_{x_i}\Delta$$

$$y_{i+1} = y_i + V_{y_i}\Delta$$

CO-STATE EQUATIONS $\bar{\lambda}^T_i = H_{\bar{x}}(i)$

$$\bar{\lambda}^T_i = \bar{\lambda}^T_{i+1}\bar{f}_{\bar{x}}\, i = \begin{bmatrix}\lambda^x_{i+1} & \lambda^y_{i+1}\end{bmatrix}\begin{bmatrix}\frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y}\\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y}\end{bmatrix}$$

$$\hookrightarrow \begin{Bmatrix}\lambda^x_i = \lambda^x_{i+1}\frac{\partial f_1}{\partial x} + \lambda^y_{i+1}\frac{\partial f_2}{\partial x}\\ \lambda^y_i = \lambda^x_{i+1}\frac{\partial f_1}{\partial y} + \lambda^y_{i+1}\frac{\partial f_2}{\partial y}\end{Bmatrix}$$

$$\longrightarrow \lambda^x_i = \lambda^x_{i+1}$$

$$\lambda^y_i = \lambda^x_{i+1}\frac{\partial V_{x_i}}{\partial y} + \lambda^y_{i+1}$$

$$\hookrightarrow V_x = V(b/d)_x + V_{river}(y)$$

$$\frac{\partial V_x}{\partial y} = \frac{\partial V_{river}}{\partial y} = \frac{d}{dy}\left(4V_{max}\,y(w-y)/w^2\right)$$

$$= \frac{4V_{max}}{w^2}(w-2y)$$

$$\longrightarrow \lambda^y_i = \lambda^x_{i+1}\left(\frac{4V_{max}}{w^2}(w-2y)\right) + \lambda^y_{i+1}$$

OPTIMALITY CRITERION $H_{\bar{u}(i)} = 0$

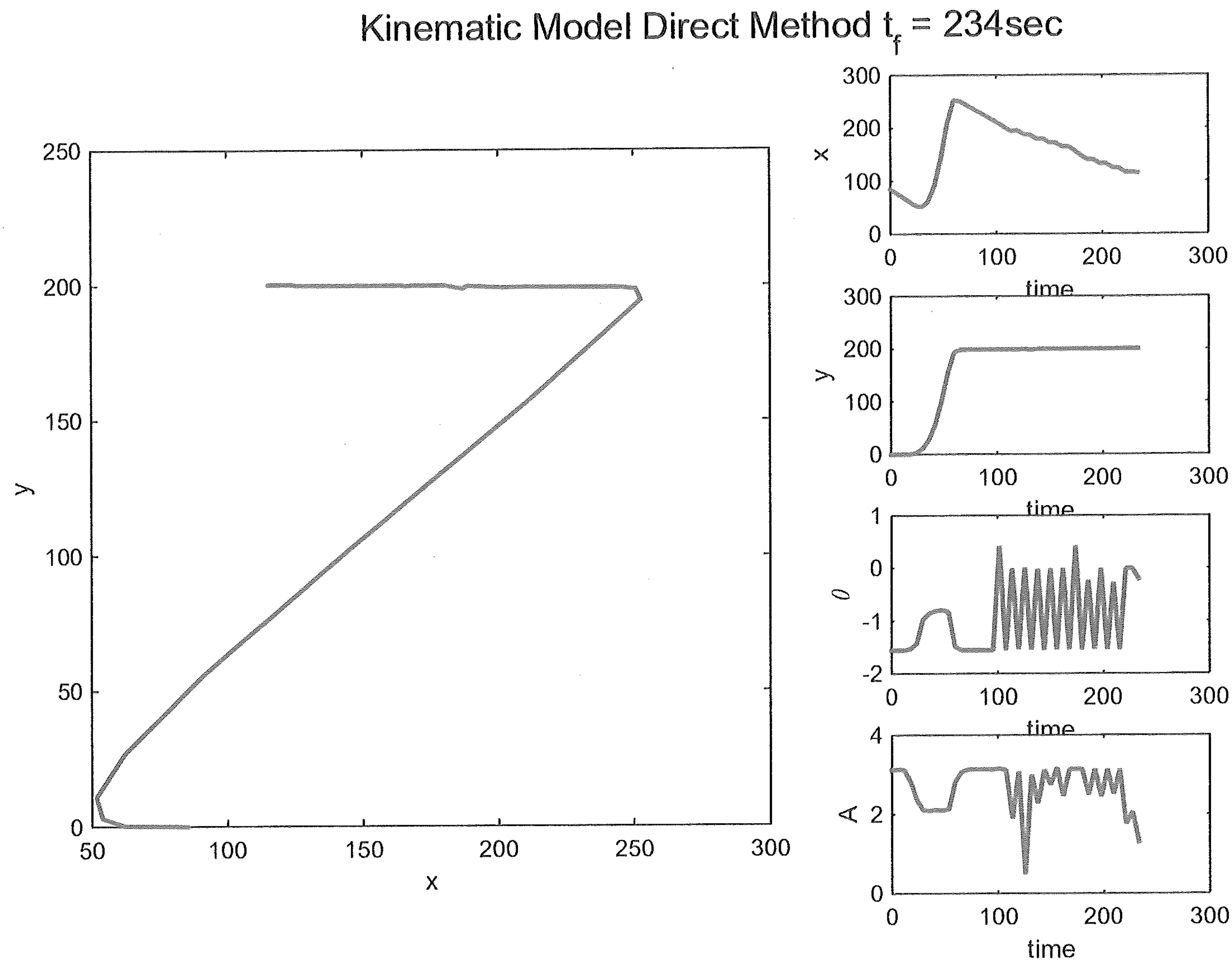
$$\bar{\lambda}^T_{i+1}\;\bar{f}_{\bar{u}}(i) = 0$$

$$\begin{bmatrix}\lambda^x_{i+1} & \lambda^y_{i+1}\end{bmatrix}\begin{bmatrix}\frac{\partial f_1}{\partial A} & \frac{\partial f_1}{\partial \theta}\\ \frac{\partial f_2}{\partial A} & \frac{\partial f_2}{\partial \theta}\end{bmatrix} = 0$$

NO CLOSED-FORM SOLUTION FOR $\bar{V}$

$$\frac{\partial f_1}{\partial A} = \frac{\partial x_i}{\partial A} + \frac{\partial V_{x_i}}{\partial A}\Delta$$

$$= 0 +$$



Kinematic Model Direct Method t_f = 234sec
250
200
150
100
50
0
y
50
100
150
200
250
300
x
300
200
100
0
x
time
y
time
θ
1
0
-1
-2
time
A
4
2
0
time

```
% Tim Coon
% Qualifying Exam Question #2, MECH 622
% 01/12/2014
% kinematic model straightforward method
clear; close all; clc;
set(0,'DefaultTextInterpreter','tex')

%------------ given ------------------------
global Vmax w Vw
Vwind = 6*0.514;             % (m/s) wind speed
Awind = deg2rad(130);        % (rad) wind direction angle wrt x-axis
Vw = [Vwind*cos(Awind); Vwind*sin(Awind)];
Vmax = 32*0.514;             % (m/s) max river velocity
w = 200;                     % (m) river width
%---final states (posiion)
xf=115;     % (m)
yf=w;       % (m)
%---initial states
x1 = 85;    % (m)
y1 = 0;     % (m)

%------------ guesses ------------------------
%---Set initial guess for final time
tf0=200;
%---Set number of time steps
N=40;
%---Set initial guess for control input
nc=2;                   % number of control inputs`
th0=-(pi/2)*ones(N,1);      % theta is the sail angle wrt -boat centerline
A0 = (atan(20/3))*ones(N,1);      % A is the heading angle wrt x-axis
% A0 = linspace(0,pi/2,N)';
% u0 = [th0; A0; tf0/N];
u0_guess = load('u0_guess.mat');
u0 = u0_guess.u;
% initial guess for states
x0 = linspace(x1,xf,N); y0 = linspace(y1,yf,N);
s0 = [x0; y0];

%------------ nlp setup ------------------------
optn=optimset('GradConst','off','Display','Iter','MaxIter',1e7,'MaxFunEvals',
1e7,'TolCon',1,'TolFun',1,'algorithm','sqp');
lb = zeros(length(u0),1); ub = zeros(length(u0),1);
lb(1:N) = -pi/2; ub(1:N) = pi/2;                % sail angle
lb(N+1:2*N) = -2*pi; ub(N+1:2*N) = 2*pi;        % heading angle
lb(2*N+1) = 100/N; ub(2*N+1) = 500/N;            % time step
[u fval] = fmincon(@Cost_MECH622_Qual,u0,[],[],[],[],lb,ub,...
                                   @Constr_MECH622_Qual,optn,N,s0);

% u = u0;
th = u(1:N);
A = u(N+1:2*N);
dt = u(2*N+1);
```

```
t = 0:dt:(N-1)*dt;
tc = 0:dt:(N-1)*dt;
s = States_MECH622_Qual(u, N, s0);
x = s(1,:);
y = s(2,:);

%% Plot results
figure(1)
Title = strcat({'Kinematic Model Direct Method t_f = '},num2str(t(end),3),{ 'sec'});
suptitle(Title)
subplot(4,6,[1 2 3 4, 7 8 9 10, 13 14 15 16, 19 20 21 22])
plot(x,y,'linewidth',2)
xlabel('x'); ylabel('y')
axis square
subplot(4,6,5:6)
plot(t,x,'linewidth',2)
xlabel('time'); ylabel('x');
subplot(4,6,11:12)
plot(t,y,'linewidth',2)
xlabel('time'); ylabel('y');
subplot(4,6,17:18)
plot(tc,th,'linewidth',2)
xlabel('time'); ylabel('\theta');
subplot(4,6,23:24)
plot(tc,A,'linewidth',2)
xlabel('time'); ylabel('A');
```

```
function [ J ] = Cost_MECH622_Qual( u, N, s0  )
%COST_MECH622_QUAL calculate value of the cost function
%   The last entry in the control vector is the time step. The cost
%   function is simply the value of time final

tf = u(end)*N;

J = tf;

end
```

```
function [ c, ceq ] = Constr_MECH622_Qual( u, N, s0 )
%CONSTR_MECH622_QUAL Contains all problem constraints
%   State Constraints
%   Initial Constraints
%   Endpoint Constraints
%   Path Constraints
%   c(x) <= 0, ceq(x) == 0

global w

% generate state vector from input guess
[s, ~, Vbr] = States_MECH622_Qual(u, N, s0);

% initial constraints
Ieq = s0(:,1) - s(:,1);

% endpoint constraints
Eeq = s0(:,end) - s(:,end);

% path constraints
% Add another constraint to prevent the force on the boat from
% being backwards. Velocity vector of the boat wrt the river must be in the
% same direction as the heading angle at all times.
% A = u(N+1:2*N);
% Vbr_angle = calcAngle(Vbr);
% Peq = [cos(A) - cos(Vbr_angle); sin(A) - sin(Vbr_angle)];

% bound the states to keep boat in the river
Py1 = -s(2,:);          % keep boat above lower bank (Py1 must be <= 0)
Py2 = s(2,:) - w;       % keep boat below upper bank (Py2 must be <= 0)

% assemble constraint vectors
% ceq = [Ieq; Eeq; Peq];          % equality constraints vector
ceq = [Ieq; Eeq];
c = [Py1'; Py2'];               % inequality constraints vector

%% nested function to calculate the heading angle wrt +x-axis
function angle = calcAngle(v)
    % calc angle between v and +x-axis=
    vx = v(1,:); vy = v(2,:);
    angle = atan(vy./vx)';
    angle(isnan(angle)) = 0;
end

end
```

```
function [s, Vvector, Vbr] = States_MECH622_Qual(u, N, s0)
% STATES_MECH622_QUAL calculates the states

x = s0(1,:);
y = s0(2,:);
th = u(1:N);
A = u(N+1:2*N);
dt = u(2*N+1);

% velocities
Vvector = zeros(2,N);
Vbr = zeros(2,N);
Vgrad = zeros(2,N);
Vguess = Vvector; Vguess(:,1) = [1;1];
Vriver = zeros(2,N);

% state propagation
for i = 1:N-1
    Vriver(1,i) = calcVriver(y(i));
    [Vvector(:,i), Vbr(:,i), Vgrad(:,i)] = calcVvector(A(i),th(i),Vguess(:,i),Vriver(:,
i));
    Vguess(:,i+1) = Vvector(:,i);       % use current velocity for next guess
    x(i+1) = x(i) + Vvector(1,i)*dt;
    y(i+1) = y(i) + Vvector(2,i)*dt;
end

s = [x; y];
```

```matlab
function [Vb,Vbr,Vb_grad] = calcVvector(A,th,Vb0,Vr,varargin)
% CALCVVECTOR calculates the absolute velocity of the boat
% Vb (2x1)= abs velocity of boat (m/s)
% Vbr (2x1) = velocity of boat wrt river
% Vw (2x1)= abs velocity of wind (m/s)
% Vr (2x1)= abs velocity of river (m/s)
% A (1x1)= heading angle of boat wrt +x-axis (rad)
% th (1x1)= angle of the main sail wrt neg boat C/L (rad)
% V0 (2x1)= initial guess for abs velocity (m/s)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% I discovered that, while the accuracy of fsolve() is heavily reliant upon
% the initial guess, fmincon() is far more robust. I made a dummy objective
% file and used an identical function for the constraints to that I used
% with fsolve.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global Vw

if nargin == 2  % then we are testing this function
    Aw = deg2rad(130);
    Vw_mag = 10*.514;
    Vw = [Vw_mag*cos(Aw); Vw_mag*sin(Aw)];
    Vr = [0; 0];
%     A = pi/2;
%     th = pi/2;
    Vb0 = [0; 0];
end
c1 = 0.5;     % wind force coeff
c2 = 0.5;     % water drag force coeff

% mu = (c1/c2)^2;                  % force coefficient
rs = [cos((A+pi)+th); sin((A+pi)+th)];    % main sail direction vector

optn = optimset('Display','off');
% [Vb_fsolve,fval] = fsolve(@V_eqns,Vb0,optn);   % does not work as well
% Vb = Vb_fsolve;

[Vb_fmincon,~,~,~,~,Vb_grad_fmincon] = ...
                      fmincon(@dummyJ,Vb0,[],[],[],[],[],[],@V_eqns2,optn);
Vb = Vb_fmincon;
Vb_grad = Vb_grad_fmincon;

Vbr = Vb - Vr;

% Vb = [Vb_fsolve,Vb_fmincon];    % for easy comparison of results


%% fsolve function (not used)
```

```
function F = V_eqns(Vb)
    Vwb = Vw - Vb;
    Vbr = Vb - Vr;
    Vbr_hat = Vbr/norm(Vbr);
    sine_alpha = det([rs';Vwb'])/norm(Vwb);
%         sine_theta = det([-Vbr_hat';rs']);
    Fs_mag = c1*norm(Vwb)^2*sine_alpha;
    Fs_br = Fs_mag*sin(th)*Vbr_hat;
    Fd = c2*norm(Vbr)*Vbr;

    F = Fs_br - Fd;
end

%% fmincon NL constraint function
% the equations herein are formed using a force balance with vector
% notation. Satisfying the force balance determines the absolute velocity
% of the boat
function [c,ceq] = V_eqns2(Vb)
    Vwb = Vw - Vb;
    Vbr = Vb - Vr;
%     Vbr_hat = Vbr/norm(Vbr);
    Vbr_hat = [cos(A); sin(A)];
    rs_X_Vwb = det([rs';Vwb']);
    sine_theta = det([-Vbr_hat';rs']);
    Fs_mag = c1*norm(Vwb)*rs_X_Vwb;
    Fs_br = Fs_mag*sine_theta*Vbr_hat;
    Fd = c2*norm(Vbr)*Vbr;

    ceq = Fs_br - Fd;
    c = [];
end

%% fmincon objective function
% this is a dummy function for fmincon() so the constraint equations can be
% solved more robustly than with fsolve()
function J = dummyJ(Vb)
    J = Vb(1);
end
end
```