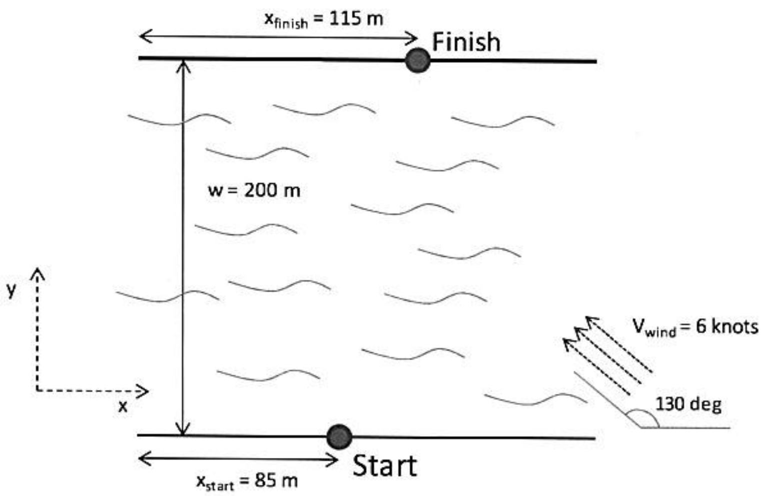
Optimal Control Qual Exam for Tim Coon
Take-Home (48 hours)

Numerical Optimization of a River Crossing:

Your task is to cross a river in minimum time using a sailboat (you may assume a point mass model). A diagram of the river crossing is shown in the below figure.



The river is flowing *from left to right* with $V_{river} = 4*V_{max}*y*(w-y)/w^2$, where V_{max} is 32 knots. Your solution to this problem must include a diagram of the crossing profile (e.g. path as a function of x and y) and a plot of the sail angle vs. time for the optimal path. You may not use GPOPS for this problem, but you may use any method within Matlab (e.g. *fmincon*, *fsolve*, etc.) that you deem appropriate. In addition to the above mentioned plots, please include:

a) A formal statement of the discretized static optimization problem that you used to solve for your optimal path. Include your objective function and ALL constraints.
b) The augmented Lagrangian function. Write out each of the terms expanded as much as possible, but leave in terms of *i*. That is, expand the state terms but not time other than the first and last time steps as is traditional.
c) The similarly expanded Hamiltonian, if you did not expand as part of b).
d) Perform an analysis (experimenting is sufficient) to determine the value of V_{max} that results in an optimal path with the minimum crossing time (an optimum of optimums, so to speak). Why is the river velocity that results in this not zero?
e) What is the minimum crossing time if you are not required to finish at a specific location on the far bank (use V_{max} = 32 knots)? Which far bank location results in this minimum crossing time?
f) *Optional:* If you solve this problem by supplying the gradients to the optimizer in Matlab, you will automatically get full credit. You must also state why your code does not require the explicit calculation of the lagrange multiplier, ν.

$\sim b/c \quad (x-x_f)+(y-y_f)=0 \equiv (x-x_f)=0 \;\&\; (y-y_f)=0$

INCLUDE ANY CODE YOU WRITE (comments are helpful)

$b/c \; (x-x_f)\leq 0 \;\&\; (y-y_f)\leq 0$

Only Dr. Cobb or Maj. Dillsaver may answer questions. Open book, (your) open notes. You may use any code from Mech 622 that you consider helpful. Do not use the internet.

V = VELOCITY OF BOAT wrt RIVER
W = VELOCITY OF WIND wrt RIVER

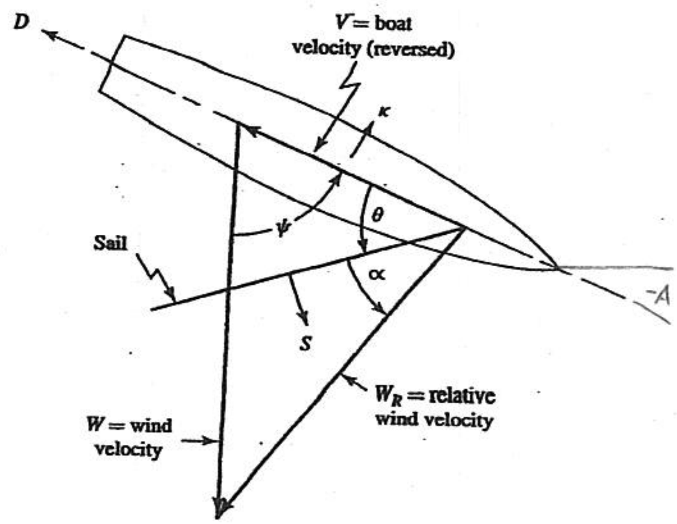


FIGURE 1.9 Force Equilibrium for a Sailboat Moving at Constant Velocity
BRYSON 1.2.7

$S = c_1 W_R^2 \sin(\alpha)$

$D = c_2 V^2$

STATES: x, y

CONTROLS: θ, A

a) FORMAL OPTIMIZATION PROBLEM STATEMENT

θ ≡ SAIL ANGLE (wrt BOAT C/L)
A ≡ HEADING ANGLE (wrt +X-AXIS)
Δ ≡ TIME STEP

MINIMIZE: $J = t_f$

SUBJECT TO:
$$x_{i+1} = x_i + (V_i \cos(A_i) + V_{river,i})\,\Delta$$
$$y_{i+1} = y_i + V_i \sin(A_i)\,\Delta$$

CONSTRAINTS: (BCs)
$$x_0 = 85m \quad y_0 = 0$$
$$x_f = 115m \quad y_f = 200m$$

$\psi(\bar{x}_N, t_f) = 0$

THE PROBLEM IS SOLVED QUASI-DYNAMICALLY. AT EACH TIME INTERVAL THE NEW POSITION IS DETERMINED BY THE PREVIOUS POSITION AND THE PREVIOUS CONTROLS WHICH DETERMINE THE VELOCITY THEN HELD CONSTANT OVER THE INTERVAL. THE VELOCITY, V, IS CALCULATED USING A FORCE BALANCE AND TRIGONOMETRY

$\mu^2 = c_1/c_2$

REF BRYSON 1.2.7

A_w = ANGLE OF WIND wrt RIVER
$\vec{\underline{W}}$ ≡ ABS VEL OF WIND

① $V^2 = \mu^2 W_R^2 \sin(\alpha)\sin(\theta)$ ← FORCE BALANCE

② $W_R \sin(\alpha + \theta) = W \sin(\psi)$

③ $W_R^2 = V^2 + W^2 - 2VW\cos(\psi)$

$\left.\begin{matrix} ② \\ ③ \end{matrix}\right\}$ TRIG

④ $\psi = A_w - A$ ← $A_w = \tan^{-1}\left(\dfrac{\underline{W}_y}{\underline{W}_x - V_{river}}\right)$

$W = [(\underline{W}_x - V_{river})^2 + \underline{W}_y^2]^{1/2}$

b) THE AUGMENTED LAGRANGIAN FUNCTION

$$\bar{J} = \Phi(\bar{x}(N), \Delta) - \bar{\lambda}^T(N)\,\bar{x}(N) + \bar{\lambda}^T(0)\,\bar{x}(0) + \sum_{i=0}^{N-1}\left[H(\bar{x}(i), \bar{u}(i), \Delta) - \bar{\lambda}^T(i)\,\bar{x}(i)\right]$$

$$\Phi(\bar{x}(N), \Delta) = \phi(\bar{x}(N), \Delta) + \bar{\nu}^T \bar{\psi}(\bar{x}(N), \Delta)$$

$$\bar{x} = \begin{bmatrix}\hat{x}\\ \hat{y}\end{bmatrix}$$

$$\bar{J} = N\Delta + \nu^{\hat{x}}\left[\hat{x}(N) - \hat{x}_f\right] + \nu^{\hat{y}}\left[\hat{y}(N) - \hat{y}_f\right] - \bar{\lambda}^T(N)\,\bar{x}(N) + \bar{\lambda}^T(0)\,\bar{x}(N) \cdots$$
$$+ \sum_{i=0}^{N-1}\left[H(\bar{x}(i), \bar{u}(i), \Delta) - \bar{\lambda}^T(i)\,\bar{x}(i)\right]$$

$d\bar{J} =$

c) EXPANDED HAMILTONIAN

$$H(i) = \bar{\lambda}^T(i+1)\,\bar{f}(i)$$

$$H(i) = \lambda^{\hat{x}}(i+1)\left[\hat{x}(i) + \left(V_i \cos(A_i) + V_{river_i}\right)\Delta\right]$$
$$+ \lambda^{\hat{y}}(i+1)\left[\hat{y}(i) + V_i \sin(A_i)\Delta\right]$$

? ASSUME $\frac{dV_i}{dy}$ NEGLIGIBLE

PARTIAL DERIVATIVES OF THE HAMILTONIAN

$$V_{river} = \frac{4V_{max}\,y(w-y)}{w^2}$$
$$= \frac{4V_{max}}{w^2}\left(wy - y^2\right)$$
$$\frac{dV_{river}}{dy} = \frac{4V_{max}}{w^2}\left(w - 2y\right)$$

$$H_x = \lambda^T(i+1)\,f_x$$
$$= \lambda_x(i+1)$$
$$H_y = \lambda_x(i+1)\,\frac{\partial V_{river}}{\partial y}\Delta + \lambda_y(i+1)$$

NO CLOSED-FORM SOLUTION FOR V, SO NO DERIVATIVES f_θ OR f_A TO FIND

$$H_\theta = ?$$

$$H_A = ?$$

$$\left[H^{\phi}_{u(i)}\right]^T = f_u^T(i)\,\lambda^{\phi}(i+1)$$

AND

$$\left[H^{\psi}_{u(i)}\right]^T = f_u^T(i)\,\lambda^{\psi}(i+1)$$

REF BRYSON

IN GENERAL. ? NOT SURE HOW THIS VECTOR IS ARRANGED FOR MULTIPLE INPUTS.

NUMERICAL SOLUTION WITH GRADIENT METHODS

FOR FREE-FINAL-TIME PROBLEMS, INCLUDE THE TIME STEP SIZE AS AN UNKNOWN ALONG WITH THE VALUES OF THE CONTROL AT EACH STATE

$$y = [u(0) \;\; u(1) \; \cdots \; u(N-1) \;\; \Delta]^T$$

① CHOOSE STEP-SIZE PARAMETERS k & m ?

② GUESS t_f AND $u(i)$ FOR $i = 0, \cdots, N-1$

d)

BECAUSE I COULD NOT CALCULATE THE GRADIENTS OF THE HAMILTONIAN FOR THE INDIRECT SOLUTION METHOD, I TRIED TO USE A MORE SIMPLE APPROACH. I MADE A COST FUNCTION, Cost_MECH622_Qual.m, TO EVALUATE THE COST AND A CONSTRAINT FUNCTION, Constr_MECH622_Qual.m, TO CALCULATE THE CONSTRAINTS. I USED fmincon.m TO TRY AND BUILD AN OPTIMAL CONTROL VECTOR WHICH INCLUDED THE STEP SIZE AS A CONTROL INPUT. UNFORTUNATELY, I NEVER OBTAINED A FEASIBLE SOLUTION.

THIS IS CLOSE TO → THE CONCEPT FOR MY RESEARCH.

WITH WORKING CODE, I COULD SIMPLY ADJUST V_{max} TO FIND THE FASTEST CROSSING TIME. I COULD EVEN USE A GRADIENT SEARCH METHOD TO FIND THE OPTIMUM OF OPTIMUMS. THE RIVER VELOCITY WILL NOT BE ZERO BECAUSE WE CAN USE THE RIVER CURRENT TO MOVE THE 30 m +X-DIRECTION AND ALSO TO INCREASE THE TOP SPEED WITHOUT TRAVELING IN -X-DIRECTION DUE TO THE WIND DIRECTION.

e)

IF MY CODE WORKED, I WOULD SIMPLY REMOVE THE Xf ENDPOINT CONSTRAINT AND CALCULATE THE RESULTING TRAJECTORY

f)

I ATTEMPTED THIS FROM THE BEGINNING. THE PROBLEM IS WITH THE CALCULATION OF THE REL VEL OF THE BOAT. BECAUSE THERE IS NO CLOSED-FORM SOLUTION TO THE SYS OF NONLINEAR EQNS, THE PARTIAL DERIVATIVES OF $f(x,u,i)$ ARE DIFFICULT TO FIND. I ASSUME A FINITE DIFF METHOD WOULD WORK, BUT I DID NOT ATTEMPT. THERE IS A JACOBIAN OPTION IN fsolve.m, BUT I DID NOT GET THIS TO WORK.

FOR V, U_r, α

THERE IS NO NEED TO DETERMINE ν EXPLICITLY B/C BOTH OF THE TERMINAL CONSTRAINTS ARE ZERO AND IT IS EQUIVALENT TO ADD ANY COMBINATION OF THEM

$$\psi_1 = x_{(N)} - x_f = 0$$

$$\psi_2 = y_{(N)} - y_f = 0$$

$$\nu^T \psi = \nu (x_{(N)} - x_f) + \nu (y_{(N)} - y_f) = 0$$

```
% Tim Coon
% Qualifying Exam Question #2, MECH 622
% 11/20/2014
% straightforward method

clear all; close all; clc;

%------------ given ----------------------
global xf yf Vmax w Vwx Vwy
Vwind = 6*0.514;            % (m/s)
Awind = deg2rad(130);       % (rad)
Vwx = Vwind*cos(Awind);     % (m/s)
Vwy = Vwind*sin(Awind);     % (m/s)
Vmax = 32*0.514;            % (m/s) max river velocity
w = 200;    % (m)
%---final states (posiion)
xf=115;     % (m)
yf=w;       % (m)
%---Set initial states [u x v y]
x0 = 85;    % (m)
y0 = 0;     % (m)
s1=[x0 y0]';
ns=length(s1);

%------------ guesses ----------------------
%---Set initial guess for final time
tf0=200;
%---Set number of time steps
N=10;
%---Set initial guess for control input
nc=2;               % number of control inputs
th0=(-pi/3)*ones(N,1)';     % theta is the sail angle wrt boat centerline
A0 = (pi/2)*ones(N,1)';     % A is the heading angle wrt x-axis


% the initial guess for the control is one-dimensional
u0 = [th0 A0 tf0/N];

% initial guess for states
s0 = zeros(ns,N);
s0(:,1) = s1;
s0(:,end) = [xf; yf];

%------------ nlp setup ----------------------

optn=optimset('GradConst','off','Display','Iter','MaxIter',100);
lb = zeros(length(u0),1); ub = zeros(length(u0),1);
lb(1:N) = (Awind-pi/2)*ones(N,1); ub(1:N) = Awind*ones(N,1);
lb(N+1:2*N) = zeros(1,N); ub(N+1:2*N) = 150;
lb(2*N+1) = 10; ub(2*N+1) = 200;
[u fval] = fmincon(@Cost_MECH622_Qual,u0,[],[],[],[],lb,ub,...
                                       @Constr_MECH622_Qual,optn,N,s0);
th = u(1:N);
A = u(N+1:2*N);
dt = u(2*N+1);

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

```
y = s(2,:);

%% Plot results
figure(1)
plot(x,y,'linewidth',2)
xlabel('x'); ylabel('y')
```

```
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

c = [];     % no inequality constraints

x = s0(1,:);
y = s0(2,:);
th = u(1:N);
A = u(N+1:2*N);
dt = u(2*N+1);
ns = min(size(s0));
v = [th; A];

chi = [-5; 2; 1];     % [V0 Wr0 alpha0] for calcV
% state constraints
V = zeros(N,1); Vriver = zeros(N,1);
for i = 1:N-1
    Vriver(i) = calcVriver(y(i));
    chi(:,i+1) = calcV(v(:,i),Vriver(i),chi(:,i));
    V(i) = chi(1,i+1);
    x(i+1) = x(i) + (-V(i)*cos(A(i))+Vriver(i))*dt;
    y(i+1) = y(i) + -V(i)*sin(A(i));
end
s = [x; y];

% initial constraints
Ieq = zeros(ns,1);
for i = 1:ns
    Ieq(i) = s0(i,1) - s(i,1);
end

% endpoint constraints
Eeq = zeros(ns,1);
for i = ns:ns
    Eeq(i) = s0(i,end) - s(i,end);
end

ceq = [Ieq; Eeq];
```

```
function s = 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);
v = [th; A];

chi = [-5; 2; 1];     % [V0 Wr0 alpha0] for calcV
% state constraints
V = zeros(N,1); Vriver = zeros(N,1);
for i = 1:N-1
    Vriver(i) = calcVriver(y(i));
    chi(:,i+1) = calcV(v(:,i),Vriver(i),chi(:,i));
    V(i) = chi(1,i+1);
    x(i+1) = x(i) + (-V(i)*cos(A(i))+Vriver(i))*dt;
    y(i+1) = y(i) + -V(i)*sin(A(i));
end
s = [x; y];
```

```
function [ Vriver ] = calcVriver( y )
%CALCVRIVER calculates the velocity of the river
%   Detailed explanation goes here

global Vmax w

% Vriver = 0;
Vriver = 4*Vmax*y*(w-y)/w^2;

end
```

```
function V = calcV(u,Vriver,chi0)
% calcV holds the set of equations to calculate relative velocity of the
% boat and other stuff using fsolve
% u = inputs
% Vriver = velocity of the river (x-direction)
% chi0 = initial guess for chi = [v Wr alpha]

global Vwx Vwy

th = u(1);
A = u(2);
Aw = atan(Vwy/(Vwx-Vriver));
psi = Aw - A;
W = sqrt((Vwx-Vriver)^2 + Vwy^2);

% guess a value of mu (will depend on boat properties)
mu = 1;         % coefficients

options = optimset('Display','off');
[X fval] = fsolve(@V_Wr_alpha_eqns,chi0,options);
V = X(1);

%% nested function

    function F = V_Wr_alpha_eqns(x)
        v = x(1);
        Wr = x(2);
        alpha = x(3);

        F = [mu^2*Wr^2*sin(alpha)*sin(th) - v^2;
             W*sin(psi) - Wr*sin(alpha+th);
             v^2 + W^2 - 2*v*W*cos(psi) - Wr^2];
    end
end
```