ME-SE-EC 710 SYLLABUS

Boston University College of Engineering

SE710 meeting with EC and ME710: Dynamic Programming and Stochastic Control

Spring 2011, MW 10-12:00 PM, PSY B35

Prof. Michael Caramanis, 15 St. Mary's Street, Rm. 137, tel. (617) 353-3247; mcaraman@bu.edu

TEXT: The required text for this course is Dynamic Programming and Optimal Control, Volumes I and II by D. Bertsekas. Volume II is formally required, but, if cost is an issue, you can get by with Volume I and class notes to be distributed regularly by email. Optimization Over Time by P. Whittle is recommended for those students interested in a more extensive and rigorous coverage of certain topics. A good presentation of the background material for the course, beyond the review that will be given in class, can be found in Luenberger's Introduction to Dynamic Systems and any good Probability text covering Derived Probability Distributions (or functions of random variables), Bayes' Theorem and Conditional Probability. Class notes distributed to all class participants by email will duplicate or complement material in the above sources.

HOMEWORK, EXAMS, PROJECT, AND GRADING: There will be homework assignments, a late midterm examination and a project (preferably but not necessarily involving computer programming) all of which will determine the final grade. Projects can be selected either in the area of a computer implementation of the analysis tools developed in the course (to verify and elaborate theoretical results, investigate the efficiency, feasibility, convergence, and accuracy of alternative numerical solutions/approximations, etc.) or in the area of applying analysis tools to a real life problem. For students exploring the possibility of taking up doctoral work in this area, the project may be viewed as an opportunity to test their interest in this field. Judgment will be exercised in determining the final grade, since use of a *formula* is not appropriate in advanced courses such as this one.

OBJECTIVES: The main objective of the course is to present a unified approach to *Markovian Decision theory and Dynamic Programming*. Applications will cover Operations Research, Stochastic Control and Computational Methods. New directions in using *Neuro-Dynamic Programming* (more simply known as *Approximate Dynamic Programming)* techniques to determine near-optimal policies for the control of otherwise intractable stochastic dynamic systems will be introduced in class and may be explored further by those interested in individual student projects. Continuous time stochastic control with Generalized Stochastic Markovian Process jump disturbances will be introduced and uniformization solution techniques will be presented. Some notable application examples in communication systems and flow control of production/service systems will be discussed.

Attention: Spring 2011 Special Dates!!!

- **Wed. April 13 Midterm Examination**
- **Mon. May 9, Projects due**

MATERIAL COVERAGE

Lectures	Coverage
1	-Introduction
2	-Sequential Decisions with Perfect Information, the D. P. algorithm, Computational Advantages.
1	-Problems with lags, auto-correlation. Reduction to standard form.
2	-Linear Systems with Quadratic Cost and the certainty equivalence principle.
2	-Inventory Control and Optimal Stopping Problems.
4	-Imperfect State Information Problems, Finite State Markov Chains, Separation of Estimation and Control in Linear Quadratic Gaussian problems (Kalman filter).
1	-System identification, dual control and adaptive control, an introduction.
2	-Discounted cost Infinite Horizon Problems and Steady State Markovian Decisions. Computational techniques (Value and Policy Iteration and Linear Programming).
2	-Average cost Infinite horizon problems. Average cost per period, steady state probabilities and the differential cost based Bellman equation. Existence of Value Function and Computational Techniques (Value and Policy Iteration, and Linear Programming).
2	-Uniformization and continuous time or fluid model approximations. Differences in the Value Iteration algorithms between the Embedded and Uniformized versions of the markovian decision making problem.
2	Applications: a) Yield Management in Bandwidth connection pricing b) Flow Control Problems arising in Communication Networks and Flexible Manufacturing Systems, c) Routing and scheduling control in multi-class queueing networks.
1	-Examination
1	-Neurodynamic Programming. The use of Features for value function approximation. The conventional numerical solution techniques (value/policy iteration and LP) and Simulation or Real-time system observation based policy/value iteration techniques in a Q learning environment where system dynamic details (system equation and probability laws) are not known.

PROJECT TOPIC EXAMPLES/SUGGESTIONS:

@Asset Selling Problem: Implement for various probability distributions including a user

specified histogram. Compute optimal transient and steady state policies. Study impact of discount rate, distribution and distribution parameters on policies.

@Fault Testing Problem: Generalize the recursion relationships for the piece-wise-linear representation of the cost to go function and the calculation of the critical probability values a_k. Implement on software and calculate the transient policy critical probabilities as well as the steady state policy critical probability for various parameter values.

@Linear System With Quadratic Costs: Generalize the optimal controller design and cost to go function formulae we derived in class by removing all the convenience assumptions. In particular derive the recursion relationships and implement them in computer software for:

$Ew \neq 0$; and $g_k = x_k' Q_k x_k + u_k' R_k u_k + x_k' D_k u_k + x_k' e_k + u_k' p_k + r_k$ where D_k, e_k, p_k, r_k are appropriate dimension constants. Do convexity conditions continue to hold? If so under what conditions?

@Imperfect State Information Problem with Finite/Discrete State and Control Spaces, Transient Behavior. Read related literature including Smallwood-Sondik paper and generalize the two dimensional examples worked out in class for a general state transition probability scheme $p_{ij}(u)$. Then implement a three dimensional problem ala Smallwood and Sondik.

@Imperfect State Information Problem with Finite/Discrete State and Control Spaces, Steady State Behavior. Implement steady state solution techniques for the n-dimensional problem (policy iteration, value iteration, Linear Programming solution).

@Linear Quadratic Gaussian Problem. Review literature and report your findings with detailed derivations of estimation formulae. Implement the discrete time Kalman filter on computer software.

@Functional approximations of the value function: quadratic function, polynomial function, and neural network functional approximations. Literature review and one selected application, coded and tested, will be the minimum expected effort.

@Flow Control Problem. Various applications and extensions of the problem discussed in class. Estimation of optimal policy parameters or optimal setup time determination are examples of possible projects in this area. Policies featuring a *hedging curve* rather than a *hedging point* are of particular interest here. They arise when the average capacity set includes the average demand vector, but individual capacity sets include only a subset of the coordinates of the demand vector.

@Review of the computational techniques applicable to infinite horizon problems as presented in Bertsekas' Dynamic Programming textbook, Chapters 1 and 4 of volume II with emphasis on sections 1.3 (Value iteration -or successive approximation- and policy iteration with Adaptive Aggregation), 4.2 and 4.3. (Optimality Conditions and Value Iteration). Think about the use of adaptive aggregation in the context of average cost per period value iteration (successive approximation) and describe how it can be adapted to the average cost per period problem.

@Review work on formulating the set-up and flow control problem (one production stage, many machines, many part types) with exponential machine failure -repair times and exponential set-up change times (Junjie Hu's Ph.D. Thesis research material will be a good start). In particular:

1. Review state discretization techniques and how the continuous state HJB equation is

converted to a discretized state equation. Note how some state transitions are disregarded (relative to the discrete time formulation) by being associated with second order terms.

2. Show how, in the average cost per period formulation, the left or right side derivative approximation can result in a discretized HJB equation which indicates that the average cost per period estimate affects the optimal control selected in the intermediate steps of the value iteration method. Resolve this by using the discussion in class and class notes on infinite horizon problems.

3. Describe how the state space can be bounded more effectively by extrapolating the value function estimates at the boundary. Apply different extrapolation techniques and test their behavior in the context of a specific numerical example.

4. Describe and speculate about parallel computation applied on numerical solution techniques in the context of value iteration approaches (serial Gauss-Seidel versus simultaneous update of the value function) and policy iteration approaches (serial versus simultaneous estimation of tentative control for each state).

5. Can neural network based approaches be used effectively?

@Review Paul Schweitzer's work on Markovian decision making system state aggregation techniques, Sethi's state aggregation paper, and Willsky's structural decomposition of multiple time scale Markov processes (papers to be supplied). Try to extract the results and solution techniques that are relevant to the policy/value iteration numerical solution techniques.

@Q learning:

1. For a discrete state and control space problem estimate the discounted cost value function J_π associated with a given policy@π by observing the dynamics of the system under policy π. (simulate the observations). Verify that the solution you get this way (is this a version of a probabilistic Gauss Seidel method?) is the same as the one you would get through the usual value or policy iteration method where the value function is evaluated at each state once in each iteration, i.e. all states are examined an equal number of times each.

2. Estimate the Q function, Q(x,u), i.e. a function that depends on state as well as control variables. Observe the dynamics of the system under each state (loop over states) and under each policy (loop over policies) allowed at time t, assuming that the system will follow the *tentatively best* policy from time t+1 on. The iteration which must be repeated for each i=1,2,3,...,n and j=1,2,3,...m, and which you hope will reach a fixed point (i.e. will converge) is:

$$Q^{v+1}(x_i,u_j)= \alpha\,[E_w\{\, g(x_i,u_j,w) + \min_{u_k\in\{u_1,..,u_m\}|w}(1\text{-}r)\; Q^v(f(x_i,u_j,w),u_k)\}] +$$

$$(1-\alpha)\,Q^v(x_i,u_j)$$

where $0<\alpha<1$ is the learning rate and r the discount rate. Repeat for α=0.1, α=0.5, and α=0.99 and draw conclusions on impact of the learning rate on the rate of convergence.

3. Repeat 1 and 2 using a functional approximation (for example a quadratic function of the state vector) for the J_π and Q functions. The update now is on the parameters of the

functional approximation. These parameters can be updated using the gradient of the mismatch with respect to the parameters and a step size α. The mismatch in 1 is:

$\{ g(x_i,u_j,w) + (1-r) J_\pi^\nu(x_{next})\} - J_\pi^\nu (x_i,u_j)$

where w and x_{next} are observed (i.e. simulated) and the superscript ν indicates the functional approximation parameter values after the ν^{th} observation. Note that the estimator of J_π does not know anything about the dynamics of x or the probability distribution of w. In other words there is a separation of the simulator of the system and the estimator of J_π. Can you think of situations where the dynamics of a system are unknown? Does this appear to be an implicit system identification procedure?

The mismatch in 2 is:

$E_w \{ g(x_i,u_j,w) + \min_{u_k \in \{u_1,..,u_m\}|w} (1-r)\ Q^\nu (f(x_i,u_j,w),u_k)\} - Q^\nu (x_i,u_j)$

where the superscript ◎ indicates the neural network parameters after the ν^{th} iteration. Note that the functional approximation used in 1 will have a smaller domain (the state space as opposed to the state×control space in 2) and hence fewer parameters to be estimated than 2.

4. Repeat the estimation of the parameters of the functional approximation of the Q function above by simply simulating the dynamics of the system and updating the parameter values after each simulation step. Do you observe that the step size ◎ selected is crucial? How is this related to the learning rate in neural network literature? How is back propagation in the neural network literature related to the gradient that you used in 3 and 4?

@Maintenance scheduling of interconnected failure prone components of a machine. Explore numerically the optimal joint maintenance policy for 2 or more components of the same machine with independent but Erlang distributed times to failure (i.e. NOT exponential!). Assume a cost structure which has the following form:

#replacement cost is highest if a component is replaced after it fails (say RF=100 units)

#replacement cost is lower if a component is replaced before it fails and the replacement is responsible for stopping the machine (say R=10 units)

#replacement cost is lowest if a component is replaced before it fails and while some other component is being replaced (say RS=5 units).

To illustrate consider a two component machine's maintenance costs for each of the possible events/control actions:

Event	Cost
1. Comp. 1 replaced upon failure	100=RF
2. Comp. 2 replaced upon failure	100=RF
3.Comp. 1 replaced upon failure, Comp. 2 replaced preventively	105=RF+RS
4.Comp. 2 replaced upon failure, Comp. 1 replaced preventively	105=RF+RS
5.Comp. 1 replaced preventively	10=R
6.Comp. 2 replaced preventively	10=R
7.Comp. 1 and Comp. 2 replaced preventively	15=R+RS

Formulate the problem for each of the following two alternative assumptions:
Assumption 1: the Erlang states are observable
Assumption 2: the Erlang states are not observable, and the only information about the hazard rate of a component is its age, i.e. the time elapsed since the last replacement.
@Use of iteratively determined features assigned to subsets of states for piece-wise functional approximation of the value function in infinite horizon problems with finite optimal control choices.

- Consider features that are constant on all states associated with the same control.
- Determine a tentative policy and use it to assign features, in effect to aggregate states to same control action subsets.
- Reevaluate functional approximations of the value function by using piece-wise functional approximations for each constant feature state subset aggregation.

@Use an appropriate function to approximate the control policy function, or the cost to go, or the Q learning function.

- When approximating the control policy function in problems with a continuous n dimensional state space and a finite control space, the problem reduces to the approximation of a partition of the n dimensional state space with multiple levels of linear inequalities (or neurons) to a mutually exclusive and exhaustive regions, each of which corresponds to a single control choice.
- When approximating the cost to go or the Q function, or when the control space is not finite, more conventional function selection may be preferable.

@Model and investigate the optimal routing and production scheduling of the multiple part type manufacturing queueing network problem introduced in class notes.
@Model and investigate numerically the bandwidth yield management problem introduced in class for controlling access to a finite capacity communication network. Investigate extending it to 2 classes of customers where only one class is responding to prices while the other has free access with constant and known exponential arrival and departure rates.
@Model the multiple class bandwidth yield management problem above with users requiring the same capacity across classes and keeping it for an exponentially distributed amount of time with the same class independent mean. Note that the system state is now the overall capacity in use at time k regardless of the class composition of users. Develop a policy iteration algorithm to solve numerically for the optimal dynamic state-feedback policy. Analyze the results and see whether they can be closely approximated by a class specific threshold type admission policy.
@Consider a modified bandwidth yield management problem where the arrival AND departure rate are functions of the price u(t) and the revenue rate over Δt equals
(the usage price charged at time t) x (the number of customers connected at time t) while both the arrival AND the departure rates, $\lambda(t)$ and $\mu(t)$ are a linear – not the same but a linear -- function of the price charged u(t).
@Investigate the performance of the multiple part type manufacturing queueing network problem introduced in class under a family of control rules known as "attractor policies". Obtain the range of possible attractor values by solving the exact LP formulation of a "small

enough" problem — see Ekbote thesis — and search for the best attractor value.
@Consider implementing a code for the infinite-horizon scheduling problems introduced in class. Use it to evaluate the various policy iteration algorithms described in class notes.
@Model and compute the optimal policy in a tandem production system with 2 machines and a demand process. Compare the optimal policy that you obtain numerically to an echelon policy, namely a policy in which Machine i produces only when the total downstream inventory is smaller than a threshold.
@Solve numerically the optimal Plug in Hybrid Electric vehicle market bid problem for 2 to 3 departure classes. See problem definition in class notes.