APPROXIMATE DYNAMIC PROGRAMMING

LECTURE 2

LECTURE OUTLINE

- Review of discounted problem theory
- Review of shorthand notation
- Algorithms for discounted DP
- Value iteration
- Various forms of policy iteration
- Optimistic policy iteration
- Q-factors and Q-learning
- Other DP models - Continuous space and time
- A more abstract view of DP
- Asynchronous algorithms

DISCOUNTED PROBLEMS/BOUNDED COST

- Stationary system with arbitrary state space

$$x_{k+1} = f(x_k, u_k, w_k), \qquad k = 0, 1, \ldots$$

- Cost of a policy $\pi = \{\mu_0, \mu_1, \ldots\}$

$$J_\pi(x_0) = \lim_{N \to \infty} \mathop{E}_{\substack{w_k \\ k=0,1,\ldots}} \left\{ \sum_{k=0}^{N-1} \alpha^k g\big(x_k, \mu_k(x_k), w_k\big) \right\}$$

with $\alpha < 1$, and for some M, we have $|g(x,u,w)| \leq M$ for all (x,u,w)

- Shorthand notation for DP mappings (operate on functions of state to produce other functions)

$$(TJ)(x) = \min_{u \in U(x)} \mathop{E}_{w} \left\{ g(x,u,w) + \alpha J\big(f(x,u,w)\big) \right\}, \; \forall\, x$$

TJ is the optimal cost function for the one-stage problem with stage cost g and terminal cost αJ.

- For any stationary policy μ

$$(T_\mu J)(x) = \mathop{E}_{w} \left\{ g\big(x, \mu(x), w\big) + \alpha J\big(f(x, \mu(x), w)\big) \right\}, \; \forall\, x$$

"SHORTHAND" THEORY – A SUMMARY

- Bellman's equation: $J^* = TJ^*$, $J_\mu = T_\mu J_\mu$ or

$$J^*(x) = \min_{u \in U(x)} \mathop{E}_{w} \Big\{ g(x,u,w) + \alpha J^*\big(f(x,u,w)\big)\Big\}, \ \forall\, x$$

$$J_\mu(x) = \mathop{E}_{w} \Big\{ g\big(x,\mu(x),w\big) + \alpha J_\mu\big(f(x,\mu(x),w)\big)\Big\}, \ \forall\, x$$

- Optimality condition:

$$\mu\text{: optimal} \quad <==> \quad T_\mu J^* = TJ^*$$

i.e.,

$$\mu(x) \in \arg \min_{u \in U(x)} \mathop{E}_{w} \Big\{ g(x,u,w) + \alpha J^*\big(f(x,u,w)\big)\Big\}, \ \forall\, x$$

- Value iteration: For any (bounded) J

$$J^*(x) = \lim_{k \to \infty} (T^k J)(x), \qquad \forall\, x$$

- Policy iteration: Given μ^k,
  - Find J_{μ^k} from $J_{\mu^k} = T_{\mu^k} J_{\mu^k}$ (policy evaluation); then
  - Find μ^{k+1} such that $T_{\mu^{k+1}} J_{\mu^k} = TJ_{\mu^k}$ (policy improvement)

MAJOR PROPERTIES

- Monotonicity property: For any functions J and J' on the state space X such that $J(x) \leq J'(x)$ for all $x \in X$, and any μ

$$(TJ)(x) \leq (TJ')(x), \quad (T_\mu J)(x) \leq (T_\mu J')(x), \;\; \forall\, x \in X$$

- Contraction property: For any bounded functions J and J', and any μ,

$$\max_x \big|(TJ)(x) - (TJ')(x)\big| \leq \alpha \max_x \big|J(x) - J'(x)\big|,$$

$$\max_x \big|(T_\mu J)(x) - (T_\mu J')(x)\big| \leq \alpha \max_x \big|J(x) - J'(x)\big|$$

- Compact Contraction Notation:

$$\|TJ - TJ'\| \leq \alpha \|J - J'\|, \;\; \|T_\mu J - T_\mu J'\| \leq \alpha \|J - J'\|,$$

where for any bounded function J, we denote by $\|J\|$ the sup-norm

$$\|J\| = \max_x \big|J(x)\big|$$

THE TWO MAIN ALGORITHMS: VI AND PI

- Value iteration: For any (bounded) J

$$J^*(x) = \lim_{k\to\infty} (T^k J)(x), \qquad \forall\, x$$

- Policy iteration: Given μ^k
  - Policy evaluation: Find J_{μ^k} by solving

$$J_{\mu^k}(x) = \mathop{E}_{w} \left\{ g\big(x, \mu^k(x), w\big) + \alpha J_{\mu^k}\big(f(x, \mu^k(x), w)\big) \right\}, \ \forall\, x$$

or $J_{\mu^k} = T_{\mu^k} J_{\mu^k}$

  - Policy improvement: Let μ^{k+1} be such that

$$\mu^{k+1}(x) \in \arg \min_{u\in U(x)} \mathop{E}_{w} \left\{ g(x, u, w) + \alpha J_{\mu^k}\big(f(x, u, w)\big) \right\}, \ \forall\, x$$

or $T_{\mu^{k+1}} J_{\mu^k} = T J_{\mu^k}$

- For the case of n states, policy evaluation is equivalent to solving an $n \times n$ linear system of equations: $J_\mu = g_\mu + \alpha P_\mu J_\mu$

- For large n, exact PI is out of the question (even though it terminates finitely as we will show)

JUSTIFICATION OF POLICY ITERATION

- We can show that $J_{\mu^k} \geq J_{\mu^{k+1}}$ for all k
- Proof: For given k, we have

$$J_{\mu^k} = T_{\mu^k} J_{\mu^k} \geq T J_{\mu^k} = T_{\mu^{k+1}} J_{\mu^k}$$

Using the monotonicity property of DP,

$$J_{\mu^k} \geq T_{\mu^{k+1}} J_{\mu^k} \geq T^2_{\mu^{k+1}} J_{\mu^k} \geq \cdots \geq \lim_{N\to\infty} T^N_{\mu^{k+1}} J_{\mu^k}$$

- Since

$$\lim_{N\to\infty} T^N_{\mu^{k+1}} J_{\mu^k} = J_{\mu^{k+1}}$$

we have $J_{\mu^k} \geq J_{\mu^{k+1}}$.

- If $J_{\mu^k} = J_{\mu^{k+1}}$, all above inequalities hold as equations, so J_{μ^k} solves Bellman's equation. Hence $J_{\mu^k} = J^*$
- Thus at iteration k either the algorithm generates a strictly improved policy or it finds an optimal policy
  - For a finite spaces MDP, the algorithm terminates with an optimal policy
  - For infinite spaces MDP, convergence (in an infinite number of iterations) can be shown

OPTIMISTIC POLICY ITERATION

- Optimistic PI: This is PI, where policy evaluation is done approximately, with a finite number of VI
- So we approximate the policy evaluation

$$J_\mu \approx T_\mu^m J$$

for some number $m \in [1, \infty)$ and initial J

- Shorthand definition: For some integers m_k

$$T_{\mu^k} J_k = T J_k, \qquad J_{k+1} = T_{\mu^k}^{m_k} J_k, \qquad k = 0, 1, \ldots$$

- If $m_k \equiv 1$ it becomes VI
- If $m_k = \infty$ it becomes PI
- Converges for both finite and infinite spaces discounted problems (in an infinite number of iterations)
- Typically works faster than VI and PI (for large problems)

APPROXIMATE PI

- Suppose that the policy evaluation is approximate,

$$\|J_k - J_{\mu^k}\| \leq \delta, \qquad k = 0, 1, \ldots$$

and policy improvement is approximate,

$$\|T_{\mu^{k+1}} J_k - T J_k\| \leq \epsilon, \qquad k = 0, 1, \ldots$$

where δ and ϵ are some positive scalars.

- Error Bound I: The sequence $\{\mu^k\}$ generated by approximate policy iteration satisfies

$$\limsup_{k \to \infty} \|J_{\mu^k} - J^*\| \leq \frac{\epsilon + 2\alpha\delta}{(1-\alpha)^2}$$

- Typical practical behavior: The method makes steady progress up to a point and then the iterates J_{μ^k} oscillate within a neighborhood of J^*.

- Error Bound II: If in addition the sequence $\{\mu^k\}$ "terminates" at $\overline{\mu}$ (i.e., keeps generating $\overline{\mu}$)

$$\|J_{\overline{\mu}} - J^*\| \leq \frac{\epsilon + 2\alpha\delta}{1-\alpha}$$

Q-FACTORS I

- Optimal Q-factor of (x, u):

$$Q^*(x,u) = E\left\{g(x,u,w) + \alpha J^*(\overline{x})\right\}$$

with $\overline{x} = f(x,u,w)$. It is the cost of starting at x, applying u is the 1st stage, and an optimal policy after the 1st stage

- We can write Bellman's equation as

$$J^*(x) = \min_{u \in U(x)} Q^*(x,u), \qquad \forall\ x,$$

- We can equivalently write the VI method as

$$J_{k+1}(x) = \min_{u \in U(x)} Q_{k+1}(x,u), \qquad \forall\ x,$$

where Q_{k+1} is generated by

$$Q_{k+1}(x,u) = E\left\{g(x,u,w) + \alpha \min_{v \in U(\overline{x})} Q_k(\overline{x}, v)\right\}$$

with $\overline{x} = f(x,u,w)$

Q-FACTORS II

- Q-factors are costs in an "augmented" problem where states are (x, u)

- They satisfy a Bellman equation $Q^* = FQ^*$ where

$$(FQ)(x,u) = E\left\{ g(x,u,w) + \alpha \min_{v \in U(\overline{x})} Q(\overline{x}, v) \right\}$$

where $\overline{x} = f(x,u,w)$

- VI and PI for Q-factors are mathematically equivalent to VI and PI for costs

- They require equal amount of computation ... they just need more storage

- Having optimal Q-factors is convenient when implementing an optimal policy on-line by

$$\mu^*(x) = \min_{u \in U(x)} Q^*(x,u)$$

- Once $Q^*(x,u)$ are known, the model [g and $E\{\cdot\}$] is not needed. Model-free operation

- Q-Learning (to be discussed later) is a sampling method that calculates $Q^*(x,u)$ using a simulator of the system (no model needed)

OTHER DP MODELS

- We have looked so far at the (discrete or continuous spaces) discounted models for which the analysis is simplest and results are most powerful
- Other DP models include:
  - Undiscounted problems ($\alpha = 1$): They may include a special termination state (stochastic shortest path problems)
  - Continuous-time finite-state MDP: The time between transitions is random and state-and-control-dependent (typical in queueing systems, called Semi-Markov MDP). These can be viewed as discounted problems with state-and-control-dependent discount factors
- Continuous-time, continuous-space models: Classical automatic control, process control, robotics
  - Substantial differences from discrete-time
  - Mathematically more complex theory (particularly for stochastic problems)
  - Deterministic versions can be analyzed using classical optimal control theory
  - Admit treatment by DP, based on time discretization

CONTINUOUS-TIME MODELS

- System equation: $dx(t)/dt = f\big(x(t), u(t)\big)$
- Cost function: $\int_0^\infty g\big(x(t), u(t)\big)$
- Optimal cost starting from x: $J^*(x)$
- δ-Discretization of time: $x_{k+1} = x_k + \delta \cdot f(x_k, u_k)$
- Bellman equation for the δ-discretized problem:

$$J^*_\delta(x) = \min_u \big\{\delta \cdot g(x, u) + J^*_\delta\big(x + \delta \cdot f(x, u)\big)\big\}$$

- Take $\delta \to 0$, to obtain the Hamilton-Jacobi-Bellman equation [assuming $\lim_{\delta \to 0} J^*_\delta(x) = J^*(x)$]

$$0 = \min_u \big\{g(x, u) + \nabla J^*(x)' f(x, u)\big\}, \qquad \forall\ x$$

- Policy Iteration (informally):
  - Policy evaluation: Given current μ, solve

$$0 = g\big(x, \mu(x)\big) + \nabla J_\mu(x)' f\big(x, \mu(x)\big), \qquad \forall\ x$$

  - Policy improvement: Find

$$\overline{\mu}(x) \in \arg\min_u \big\{g(x, u) + \nabla J_\mu(x)' f(x, u)\big\}, \qquad \forall\ x$$

- Note: Need to learn $\nabla J_\mu(x)$ NOT $J_\mu(x)$

A MORE GENERAL/ABSTRACT VIEW OF DP

- Let Y be a real vector space with a norm $\|\cdot\|$
- A function $F : Y \mapsto Y$ is said to be a contraction mapping if for some $\rho \in (0, 1)$, we have

$$\|Fy - Fz\| \leq \rho\|y - z\|, \qquad \text{for all } y, z \in Y.$$

ρ is called the modulus of contraction of F.

- Important example: Let X be a set (e.g., state space in DP), $v : X \mapsto \Re$ be a positive-valued function. Let $B(X)$ be the set of all functions $J : X \mapsto \Re$ such that $J(x)/v(x)$ is bounded over x.
- We define a norm on $B(X)$, called the weighted sup-norm, by

$$\|J\| = \max_{x \in X} \frac{|J(x)|}{v(x)}.$$

- Important special case: The discounted problem mappings T and T_μ [for $v(x) \equiv 1$, $\rho = \alpha$].

CONTRACTION MAPPINGS: AN EXAMPLE

- Consider extension from finite to countable state space, $X = \{1, 2, \ldots\}$, and a weighted sup norm with respect to which the one stage costs are bounded

- Suppose that T_μ has the form

$$(T_\mu J)(i) = b_i + \alpha \sum_{j \in X} a_{ij}\, J(j), \qquad \forall\ i = 1, 2, \ldots$$

where b_i and a_{ij} are some scalars. Then T_μ is a contraction with modulus ρ if and only if

$$\frac{\sum_{j \in X} |a_{ij}|\, v(j)}{v(i)} \leq \rho, \qquad \forall\ i = 1, 2, \ldots$$

- Consider T,

$$(TJ)(i) = \min_{\mu} (T_\mu J)(i), \qquad \forall\ i = 1, 2, \ldots$$

where for each $\mu \in M$, T_μ is a contraction mapping with modulus ρ. Then T is a contraction mapping with modulus ρ

- Allows extensions of main DP results from bounded one-stage cost to unbounded one-stage cost.

CONTRACTION MAPPING FIXED-POINT TH.

- Contraction Mapping Fixed-Point Theorem: If $F : B(X) \mapsto B(X)$ is a contraction with modulus $\rho \in (0,1)$, then there exists a unique $J^* \in B(X)$ such that

$$J^* = FJ^*.$$

Furthermore, if J is any function in $B(X)$, then $\{F^k J\}$ converges to J^* and we have

$$\|F^k J - J^*\| \leq \rho^k \|J - J^*\|, \qquad k = 1, 2, \ldots.$$

- This is a special case of a general result for contraction mappings $F : Y \mapsto Y$ over normed vector spaces Y that are complete: every sequence $\{y_k\}$ that is Cauchy (satisfies $\|y_m - y_n\| \to 0$ as $m, n \to \infty$) converges.
- The space $B(X)$ is complete (see the text for a proof).

ABSTRACT FORMS OF DP

- We consider an abstract form of DP based on monotonicity and contraction

- Abstract Mapping: Denote $R(X)$: set of real-valued functions $J : X \mapsto \Re$, and let $H : X \times U \times R(X) \mapsto \Re$ be a given mapping. We consider the mapping

$$(TJ)(x) = \min_{u \in U(x)} H(x, u, J), \qquad \forall\, x \in X.$$

- We assume that $(TJ)(x) > -\infty$ for all $x \in X$, so T maps $R(X)$ into $R(X)$.

- Abstract Policies: Let $\mathcal{M}$ be the set of "policies", i.e., functions μ such that $\mu(x) \in U(x)$ for all $x \in X$.

- For each $\mu \in \mathcal{M}$, we consider the mapping $T_\mu : R(X) \mapsto R(X)$ defined by

$$(T_\mu J)(x) = H\big(x, \mu(x), J\big), \qquad \forall\, x \in X.$$

- Find a function $J^* \in R(X)$ such that

$$J^*(x) = \min_{u \in U(x)} H(x, u, J^*), \qquad \forall\, x \in X$$

EXAMPLES

- Discounted problems

$$H(x,u,J) = E\big\{g(x,u,w) + \alpha J\big(f(x,u,w)\big)\big\}$$

- Discounted "discrete-state continuous-time" Semi-Markov Problems (e.g., queueing)

$$H(x,u,J) = G(x,u) + \sum_{y=1}^{n} m_{xy}(u)J(y)$$

where m_{xy} are "discounted" transition probabilities, defined by the distribution of transition times

- Minimax Problems/Games

$$H(x,u,J) = \max_{w \in W(x,u)} \big[g(x,u,w) + \alpha J\big(f(x,u,w)\big)\big]$$

- Shortest Path Problems

$$H(x,u,J) = \begin{cases} a_{xu} + J(u) & \text{if } u \neq d, \\ a_{xd} & \text{if } u = d \end{cases}$$

where d is the destination. There are stochastic and minimax versions of this problem

ASSUMPTIONS

- Monotonicity: If $J, J' \in R(X)$ and $J \leq J'$,

$$H(x, u, J) \leq H(x, u, J'), \qquad \forall\, x \in X,\ u \in U(x)$$

- We can show all the standard analytical and computational results of discounted DP if monotonicity and the following assumption holds:

- Contraction:
  - For every $J \in B(X)$, the functions $T_\mu J$ and TJ belong to $B(X)$
  - For some $\alpha \in (0,1)$, and all μ and $J, J' \in B(X)$, we have

$$\|T_\mu J - T_\mu J'\| \leq \alpha \|J - J'\|$$

- With just monotonicity assumption (as in undiscounted problems) we can still show various forms of the basic results under appropriate conditions

- A weaker substitute for contraction assumption is semicontractiveness: (roughly) for some μ, T_μ is a contraction and for others it is not; also the "noncontractive" μ are not optimal

RESULTS USING CONTRACTION

- Proposition 1: The mappings T_μ and T are weighted sup-norm contraction mappings with modulus α over $B(X)$, and have unique fixed points in $B(X)$, denoted J_μ and J^*, respectively (cf. Bellman's equation).

Proof: From the contraction property of H.

- Proposition 2: For any $J \in B(X)$ and $\mu \in \mathcal{M}$,

$$\lim_{k\to\infty} T_\mu^k J = J_\mu, \qquad \lim_{k\to\infty} T^k J = J^*$$

(cf. convergence of value iteration).

Proof: From the contraction property of T_μ and T.

- Proposition 3: We have $T_\mu J^* = T J^*$ if and only if $J_\mu = J^*$ (cf. optimality condition).

Proof: $T_\mu J^* = T J^*$, then $T_\mu J^* = J^*$, implying $J^* = J_\mu$. Conversely, if $J_\mu = J^*$, then $T_\mu J^* = T_\mu J_\mu = J_\mu = J^* = T J^*$.

RESULTS USING MON. AND CONTRACTION

- Optimality of fixed point:

$$J^*(x) = \min_{\mu \in \mathcal{M}} J_\mu(x), \qquad \forall\, x \in X$$

- Existence of a nearly optimal policy: For every $\epsilon > 0$, there exists $\mu_\epsilon \in \mathcal{M}$ such that

$$J^*(x) \le J_{\mu_\epsilon}(x) \le J^*(x) + \epsilon, \qquad \forall\, x \in X$$

- Nonstationary policies: Consider the set Π of all sequences $\pi = \{\mu_0, \mu_1, \ldots\}$ with $\mu_k \in \mathcal{M}$ for all k, and define

$$J_\pi(x) = \liminf_{k \to \infty} (T_{\mu_0} T_{\mu_1} \cdots T_{\mu_k} J)(x), \qquad \forall\, x \in X,$$

with J being any function (the choice of J does not matter)

- We have

$$J^*(x) = \min_{\pi \in \Pi} J_\pi(x), \qquad \forall\, x \in X$$

THE TWO MAIN ALGORITHMS: VI AND PI

- Value iteration: For any (bounded) J

$$J^*(x) = \lim_{k\to\infty} (T^k J)(x), \qquad \forall\, x$$

- Policy iteration: Given μ^k
  - Policy evaluation: Find J_{μ^k} by solving

$$J_{\mu^k} = T_{\mu^k} J_{\mu^k}$$

  - Policy improvement: Find μ^{k+1} such that

$$T_{\mu^{k+1}} J_{\mu^k} = T J_{\mu^k}$$

- Optimistic PI: This is PI, where policy evaluation is carried out by a finite number of VI
  - Shorthand definition: For some integers m_k

$$T_{\mu^k} J_k = T J_k, \qquad J_{k+1} = T_{\mu^k}^{m_k} J_k, \qquad k = 0, 1, \ldots$$

  - If $m_k \equiv 1$ it becomes VI
  - If $m_k = \infty$ it becomes PI
  - For intermediate values of m_k, it is generally more efficient than either VI or PI

ASYNCHRONOUS ALGORITHMS

- Motivation for asynchronous algorithms
  - Faster convergence
  - Parallel and distributed computation
  - Simulation-based implementations

- General framework: Partition X into disjoint nonempty subsets $X_1, \ldots, X_m$, and use separate processor ℓ updating $J(x)$ for $x \in X_\ell$

- Let J be partitioned as

$$J = (J_1, \ldots, J_m),$$

where J_ℓ is the restriction of J on the set X_ℓ.

- Synchronous VI algorithm:

$$J_\ell^{t+1}(x) = T(J_1^t, \ldots, J_m^t)(x), \quad x \in X_\ell,\ \ell = 1, \ldots, m$$

- Asynchronous VI algorithm: For some subsets of times $\mathcal{R}_\ell$,

$$J_\ell^{t+1}(x) = \begin{cases} T(J_1^{\tau_{\ell 1}(t)}, \ldots, J_m^{\tau_{\ell m}(t)})(x) & \text{if } t \in \mathcal{R}_\ell, \\ J_\ell^t(x) & \text{if } t \notin \mathcal{R}_\ell \end{cases}$$

where $t - \tau_{\ell j}(t)$ are communication "delays"

ONE-STATE-AT-A-TIME ITERATIONS

- Important special case: Assume n "states", a separate processor for each state, and no delays

- Generate a sequence of states $\{x^0, x^1, \ldots\}$, generated in some way, possibly by simulation (each state is generated infinitely often)

- Asynchronous VI:

$$J_\ell^{t+1} = \begin{cases} T(J_1^t, \ldots, J_n^t)(\ell) & \text{if } \ell = x^t, \\ J_\ell^t & \text{if } \ell \neq x^t, \end{cases}$$

where $T(J_1^t, \ldots, J_n^t)(\ell)$ denotes the ℓ-th component of the vector

$$T(J_1^t, \ldots, J_n^t) = TJ^t,$$

- The special case where

$$\{x^0, x^1, \ldots\} = \{1, \ldots, n, 1, \ldots, n, 1, \ldots\}$$

is the Gauss-Seidel method

ASYNCHRONOUS CONV. THEOREM I

- KEY FACT: VI and also PI (with some modifications) still work when implemented asynchronously

- Assume that for all $\ell, j = 1, \ldots, m$, $\mathcal{R}_\ell$ is infinite and $\lim_{t\to\infty} \tau_{\ell j}(t) = \infty$

- Proposition: Let T have a unique fixed point J^*, and assume that there is a sequence of nonempty subsets $S(k) \subset R(X)$ with $S(k+1) \subset S(k)$ for all k, and with the following properties:

  (1) Synchronous Convergence Condition: Every sequence $\{J^k\}$ with $J^k \in S(k)$ for each k, converges pointwise to J^*. Moreover,

  $$TJ \in S(k+1), \qquad \forall\, J \in S(k),\ k = 0, 1, \ldots .$$

  (2) Box Condition: For all k, $S(k)$ is a Cartesian product of the form

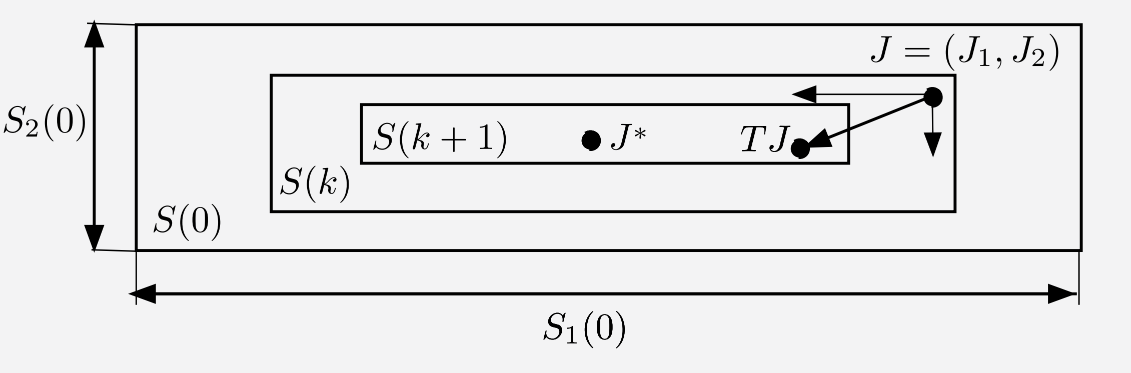
  $$S(k) = S_1(k) \times \cdots \times S_m(k),$$

  where $S_\ell(k)$ is a set of real-valued functions on X_ℓ, $\ell = 1, \ldots, m$.

Then for every $J \in S(0)$, the sequence $\{J^t\}$ generated by the asynchronous algorithm converges pointwise to J^*.

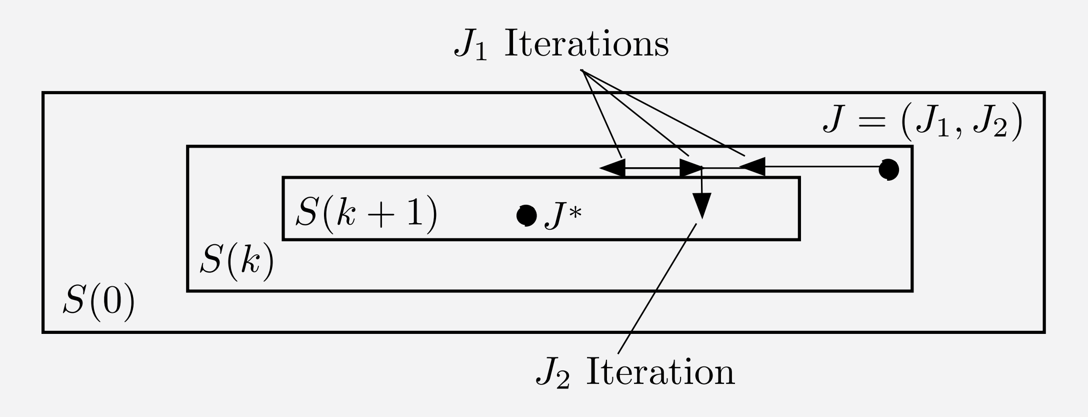
ASYNCHRONOUS CONV. THEOREM II

- Interpretation of assumptions:



A synchronous iteration from any J in $S(k)$ moves into $S(k+1)$ (component-by-component)

- Convergence mechanism:



Key: "Independent" component-wise improvement. An asynchronous component iteration from any J in $S(k)$ moves into the corresponding component portion of $S(k+1)$