Spectral learning for structured partially observable environments

Lucas Langer

McGill University

lucas.langer@mail.mcgill.ca

August 18, 2015

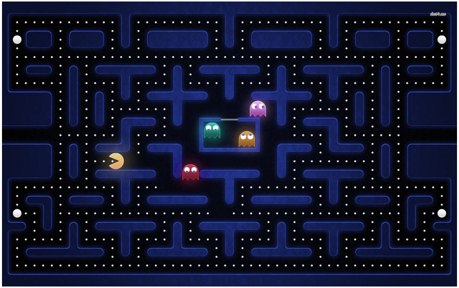
Overview

1. Predictive state representations
2. The Base System
3. Experimental results
4. Learning the Base System

Structure partially observable environments

Structured Environments

1. Goal: Predictions
2. Plan: Exploit structure
3. Example: Pacman



Structure partially observable environments

- Goal: Improve predictive performance
- Example: Pacman
- Plan: Represent structure in model

PSRs: The Timing Case

- PSRs defined by: $< \alpha_0, \{A_\sigma\}, \alpha_\infty\} >$

  α_0: Initial weighting on states $1xn$

  A_σ Transition matrix nxn

  α_∞: Normalizer $nx1$
- PSRs compute probabilities of observations

  σ represents one time unit, σ^k represents k time units

  $f(\sigma^k) = \alpha_0 * A_\sigma^k * \alpha_\infty$
- Examples of a PSRs: HMMs, POMDPs

Overview of Learning

- Spectral algorithms can learn PSRs
- Flavour of learning algorithm:

  Step 1: Represent data as a matrix

  Step 2: Singular value decomposition

  Step 3: Pick number of states for PSR

  Step 4: Learn the PSR with matrix computations

The Base System

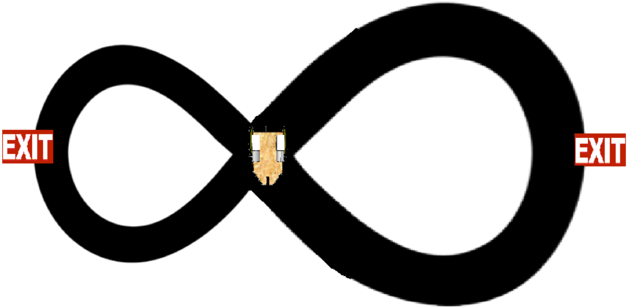
- Number representations: $11 = 2^3 + 2^1 + 2^0$
- Timing queries $f(a^1 1) = \alpha * A_(a^8) * A_(a^2) * A_(a^1)$
- Motivation:

  1) Express transitions directly to avoid error build up

  2) Faster queries. Discussion $\alpha_0 * (A_\sigma)^k$

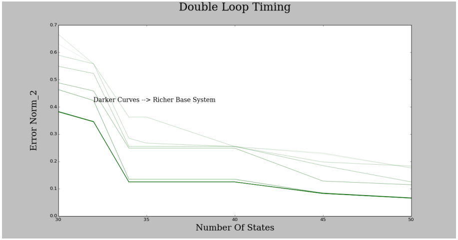
Timing with the Base

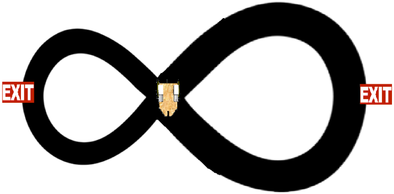
Agent goes through loops until leaving through an exit state. Exit states have transition probabilities of 0.4 and 0.6. Loop lengths are 64 and 16.



Double Loop Results

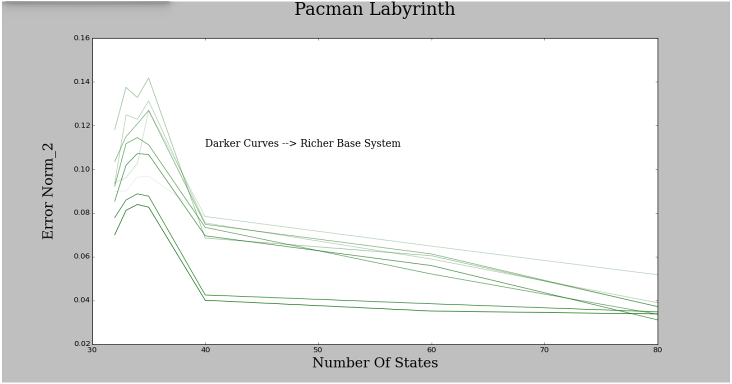
$$||f_A - f_A Bar|| = (\sum(f_A(x) - f_A Bar)^2)^(0.5)$$





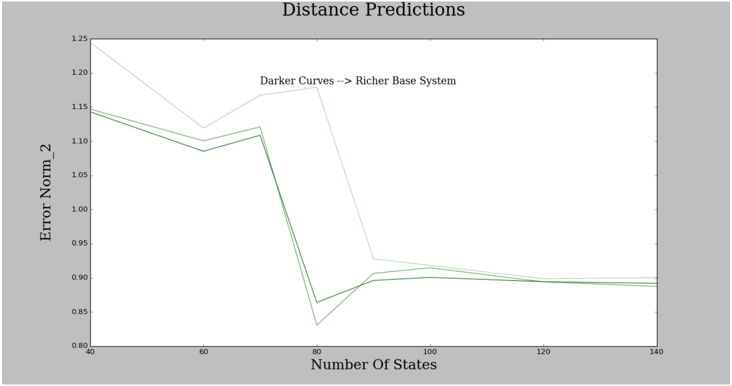
PacMan Labyrinth Results

$$||f_A - f_A Bar|| = (\sum(f_A(x) - f_A Bar)^2)^(0.5)$$



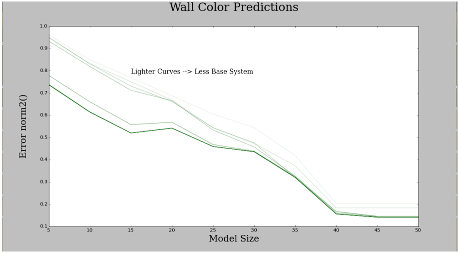
Distance Predictions

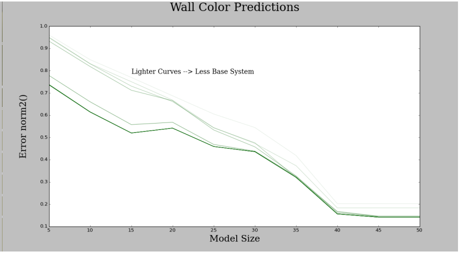
We use $\alpha_0 * A_\sigma^k$ as a representation of state. Linear regression gives us a distance weighting on states.



Wall Color Predictions

We paint the first loop green and the second loop red.





Picking the Base System

- How do we pick transition operators?

  Observations: $\{a^{30}{:}10,\ a^{60}{:}5,\ b^{18}{:}15\}$

  Desired Base System: A_a^{30}, A_b^{18}, A_a, A_b
- Substring properties: **long**, **frequent**, **diverse**
- Solution: iterative greedy heuristic

Computing with the Base System

- Using the Base System well involves requires good **string partitions**
  Query string: "abcacb", Base System $= \{A_{ab}, A_{bca}, A_{cb}, A_a, A_b\}$
  Desired partition: "a—bca—cb""
- Goal: minimize matrices used
- Solution: dynamic programming

Conclusion and Future Work

- What's left for the Base System?
  1) Theoretical analysis
  2) Test heuristics for selection and querying
  3) Further optimize heuristics

Questions? Comments?