Using Hidden Markov Models as a Statistical Process Control Technique: An Example from a ML 5 Organization

Bob Moore, Senior Principal Process Engineer, Business Transformation Institute, Inc. (BTI)

Ray Luke, Engineer, Raytheon Space & Airborne Systems

Tony Fields, Senior Software Engineer, Raytheon Space & Airborne Systems

What Is A Hidden Markov Model?

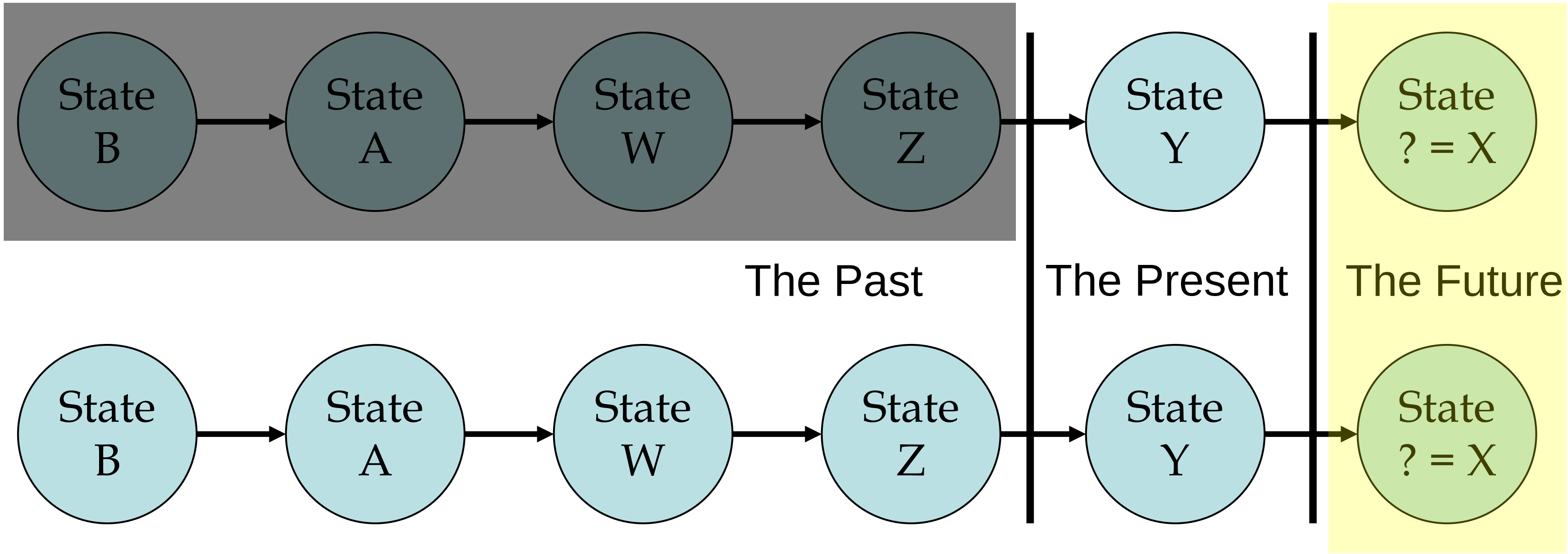
- Hidden Markov Models (HMMs) are a method for analyzing the output and underlying structure (states) of a random or seemingly random process using only the data sequences observed on the process.
- HMMs provide a fast and efficient mathematical method for analyzing and drawing conclusions about systems where the underlying processes cannot be directly observed.
- HMMs have found successful applications in speech processing, speaker recognition, cryptology, signal processing, queuing theory, coding theory, and communications systems.

Markov Chains Defined

- Let P = the probability that a process is in state “X” at time t, given that we know the state of the process at *all previous* times.
- Let Q = the probability that a process is in state “X” at time t, given that we know *only* the state of the process at previous time mark (t-1).
- A random process is a Markov chain if P = Q.

Markov Chain Illustrated

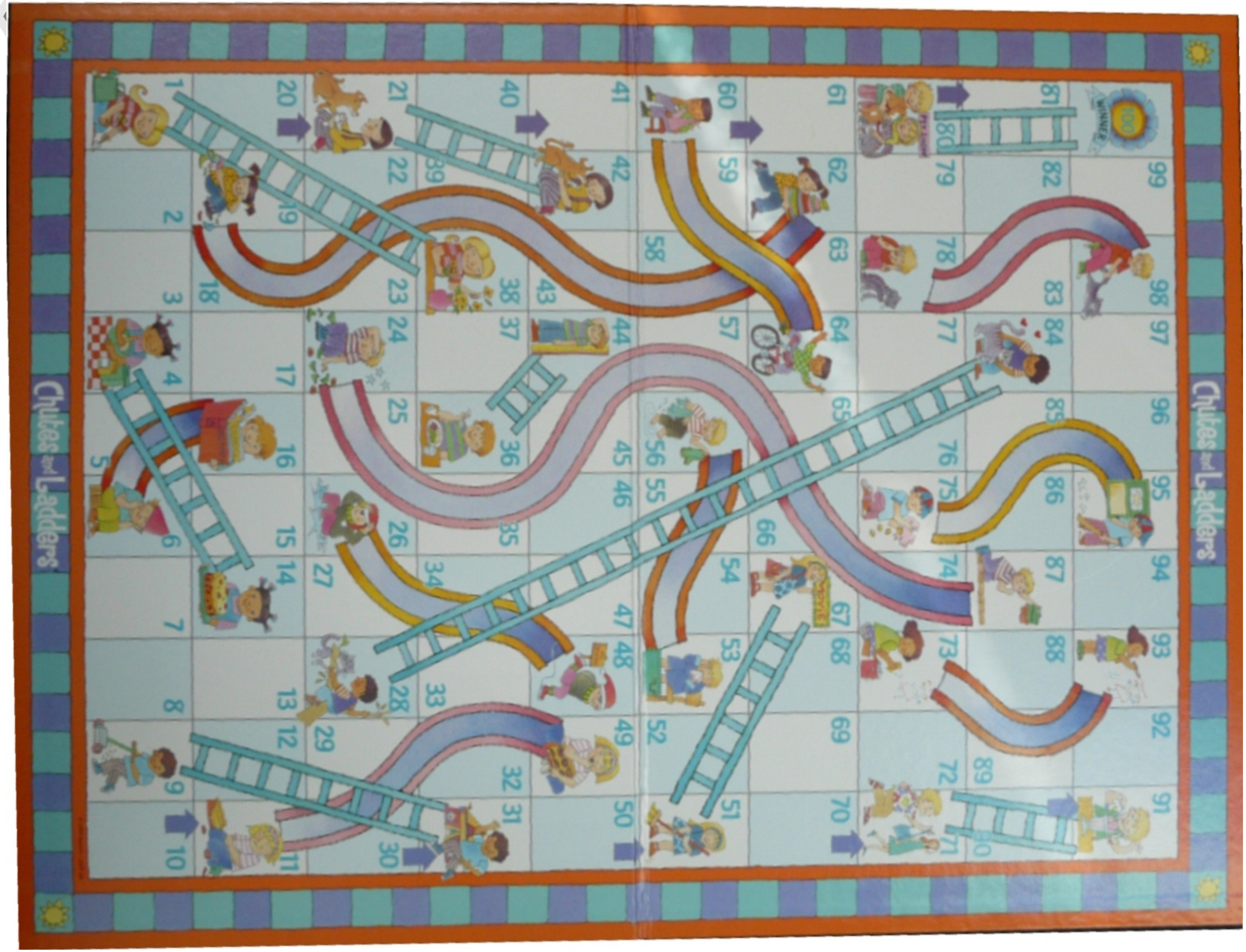
For a Markov Chain, predicting the future based on “the present” is is just as accurate as predicting the future based on both “the past” and “the present” combined.



Markov Chain Example (1)

- Remember the game *Chutes and Ladders* (sometimes “Snakes and Ladders”)?
- Chutes and Ladders represents a Markov chain: if your playing piece is on any given square, the probability of moving to any other square is fixed and independent of any previous game history.

Markov Chain Example (2)



Used here under the education fair use provision.

Markov Chain Example (3)

- Suppose we are on square 32. Then
  - Probability(next square is 33, given we're *now* on 32) =

    Probability(next square is 33, given we're *now* on 32, given that we *were* on 31 on the previous move) =

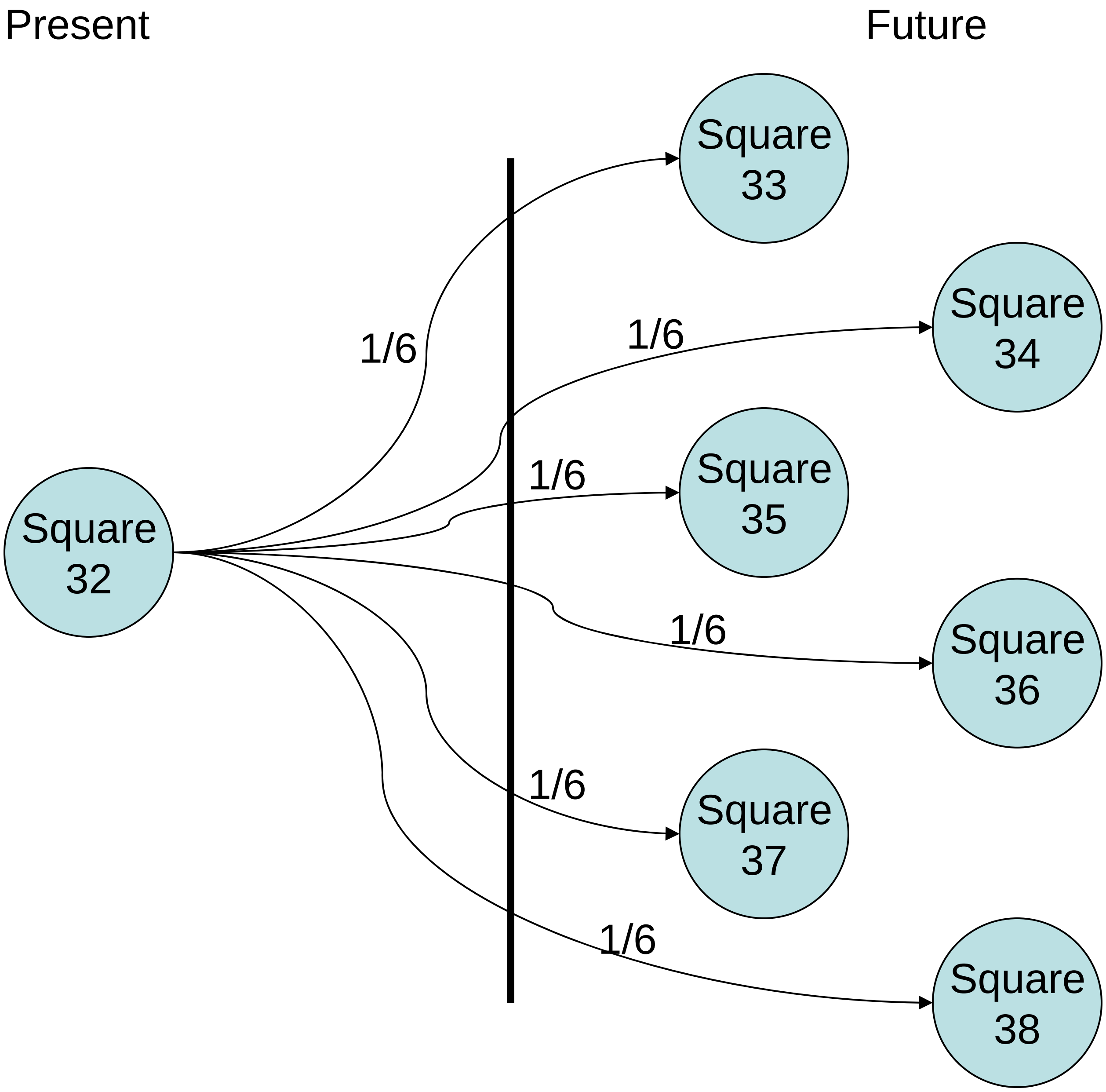
    Probability(next square is 33, given we're *now* on 32, given that we *were* on 30 on the previous move) = . . . =

    Probability(next square is 33, given we're *now* on 32, given that we *were* on 26 on the previous move) =

    1/6.
  - This is true for any square: Probability(next square is 34, given we're *now* on 32) = 1/6 = Probabililty(next square is 34, given we're *now* on 32, given that we *were* on whatever square)

Business Transformation Institute

Markov Chain Notation



Markov Chain Notation

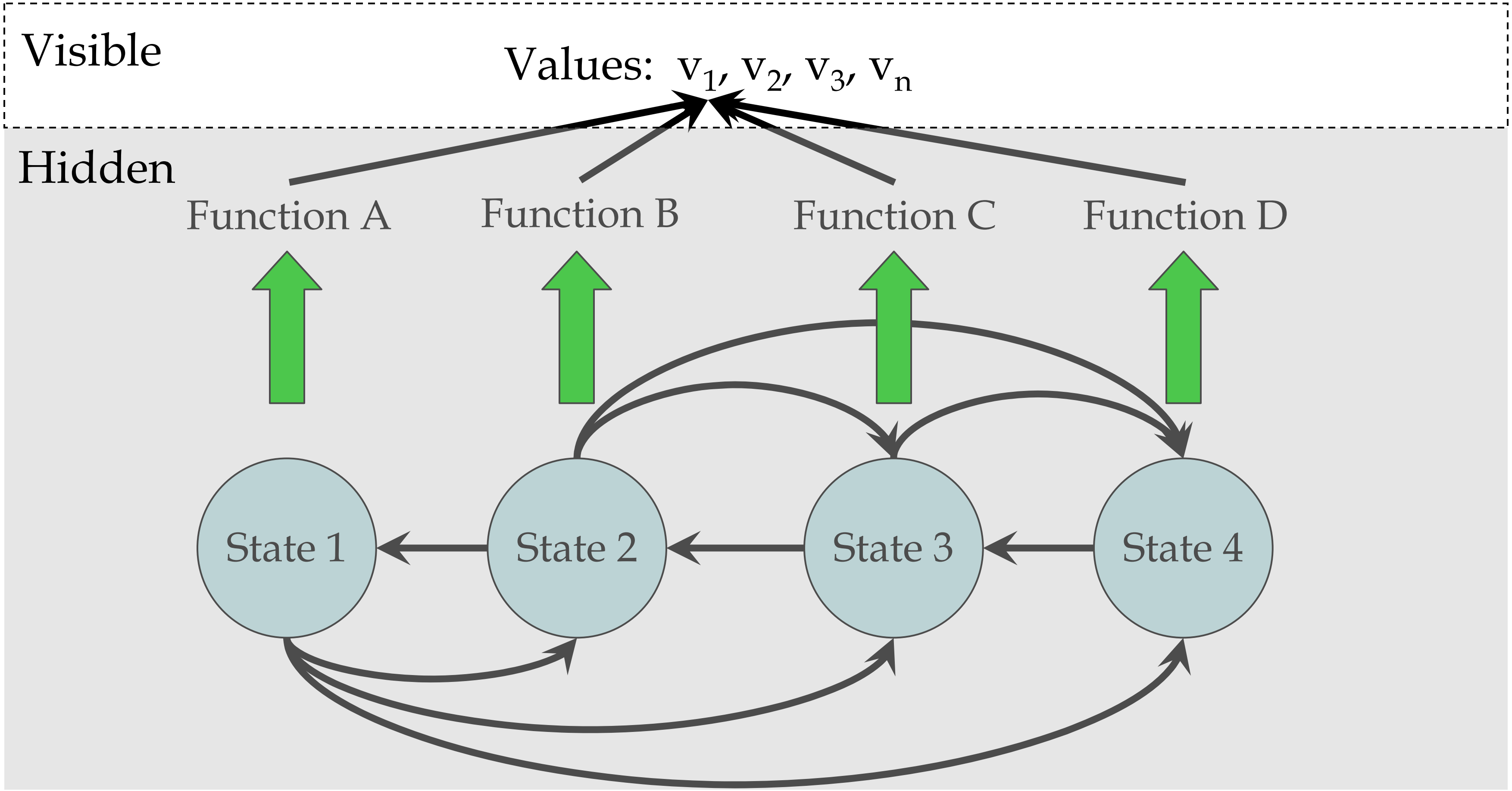
We can represent the Chutes and Ladder Markov chain as a state transition matrix T =

	1	2	3	4	5	6	7	…	38	…	100
Start	0	1/6	1/6	0	1/6	1/6	0	…	1/6	…	0
1	0	0	0	0	0	0	0	…	0	…	0
2	0	0	1/6	0	1/6	1/6	1/6	…	0	…	0
3	0	0	0	0	1/6	1/6	1/6	…	0	…	0
4	0	0	0	0	0	0	0	…	0	…	0
5	0	0	0	0	0	1/6	1/6	…	0	…	0
6	0	0	0	0	0	0	1/6	…	0	…	0
7	0	0	0	0	0	0	0	…	0	…	0
…	…	…	…	…	…	…	…	…	…	…	…
100	0	0	0	0	0	0	0	…	0	…	0

The row headings are the starting state, the column headings are the ending state in one move.

HMMs Illustrated

In a HMM, we don’t get to observe the Markov chain – only values generated by functions that are set by the Markov chain. (This is why it is called “hidden”!)



An HMM Example (1)

- Consider a slot machine!
- Every time the slot machine's "lever" is pulled, a symbol sequence is generated.

An HMM Example (2)

- We do not have any direct knowledge about the process that is generating this symbol sequence.
- The only observations that we have on the machine are the symbol sequences.
- We do know:
  - The mechanism generating the sequences assumes states which are more favorable or less favorable for a winning combination depending on how often the machine is played.
  - From our perspective, the machine's internal mechanism transitions between these states at apparently random times.
    - The progression may be structured in terms of which state passes into which state.

An HMM Example (3)

- HMMs can help us answer the following:
  - Can we determine the states of the slot machine’s internal mechanism in terms of generating symbol sequences? (Can we pick a machine that is ripe for a winner?)
  - Can we calculate the chance of seeing any particular symbol sequence or collection of symbol sequences? (Can we figure out the chance of winning big?)
  - How can we adjust our model of the slot machine so that it most accurately describes the machine’s behavior? (Can we make our predictions accurate enough and easy enough to be useful?)

What Do We Mean By "Statistical Process Control"?

- When a process is in control, process measurements appear random.
  - The random behavior will be according to one or more probability distribution functions.
  - "Common causes of variation" dominate the measurements.
- When a process is not in control, process measurements exhibit some different behavior than "in control" measurements.
  - The "not in control" behavior may be:
    - Deterministic
    - Random with a different distribution than "in control" behavior.

How Do Hidden Markov Models Apply To Statistical Process Control?

- We observe some sequence of measurement data about the process.
- The data is generated by some unknown function based on the unobserved state of the process.
- Using the sequence of measurement data, HMMs help us to calculate the state of the process.
- If the states of the process are "in control" and "not in control", then the HMM will help us recognize:
  - When only common causes of variation are present.
  - When special causes of variation are present.

HMM Calculations

- This presentation is not intended to teach the algorithms of SPC calculations using HMMs.
  - Mathematics software such as MATLAB with the “Statistics Toolbox” provides all of the tools without any programming needed.
  - Free HMM software (Java, Visual C) is available on the internet – be prepared to do some programming!
- Out-of-the box MATLAB HMM routines were supplemented with:
  - For modeling normal output distributions: HMM Toolbox for MATLAB by Kevin Murphy – http://people.cs.ubc.ca/~murphyk/Software/HMM/hmm.html

HMM Application: The Data Set

- The data are provided by Raytheon Space and Airborne Systems, Intelligence Surveillance and Reconnaissance Systems.
  - The software programs operating in the Dallas, Texas area for Raytheon SAS/ISRS were appraised as Maturity Level 5 in 2007.
- The data consists of software code defect data from peer reviews of 47 software development projects.
- A total of 867 data points are available.

Data Model Setup: Notation

- Let $D_{i,j}$ = defects reported from the i^{th} peer review of the j^{th} project.
- Let $d_{i,j}$ = actual defects in the code corresponding to the i^{th} review of the j^{th} project.
- Let $e_{i,j}$ = errors introduced by the reviewers in the i^{th} review of the j^{th} project
  - Defects reported that do not exist.
  - Defects missed that do exist.
- Let C_j = a matrix of properties of the code of the j^{th} project (e.g., lines of code, cyclomatic complexity)
- Let $\varepsilon_{i,j}$ = unknown random effect applying to the i^{th} review of the j^{th} project.

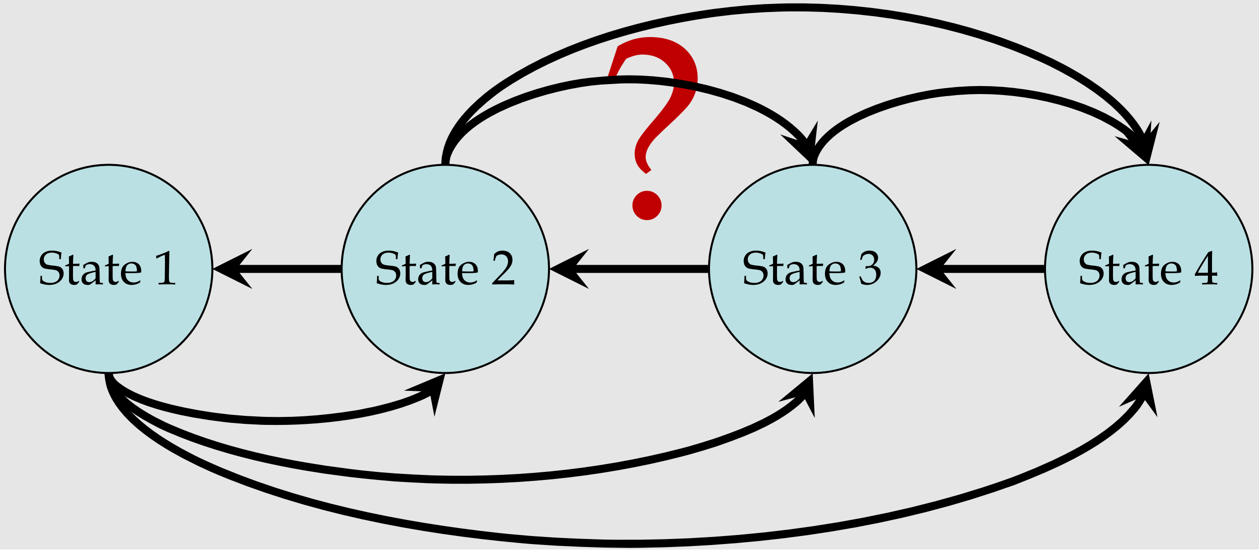
Data Model Setup

Visible

For some project $\boldsymbol{j}$: $D_{1,\boldsymbol{j}}$, $D_{2,\boldsymbol{j}}$, . . ., $D_{n,\boldsymbol{j}}$.

Hidden

$d_{1,\boldsymbol{j}}+e_{1,\boldsymbol{j}}$ $d_{2,\boldsymbol{j}}+e_{2,\boldsymbol{j}}$ … $d_{n,\boldsymbol{j}}+e_{n,\boldsymbol{j}}$



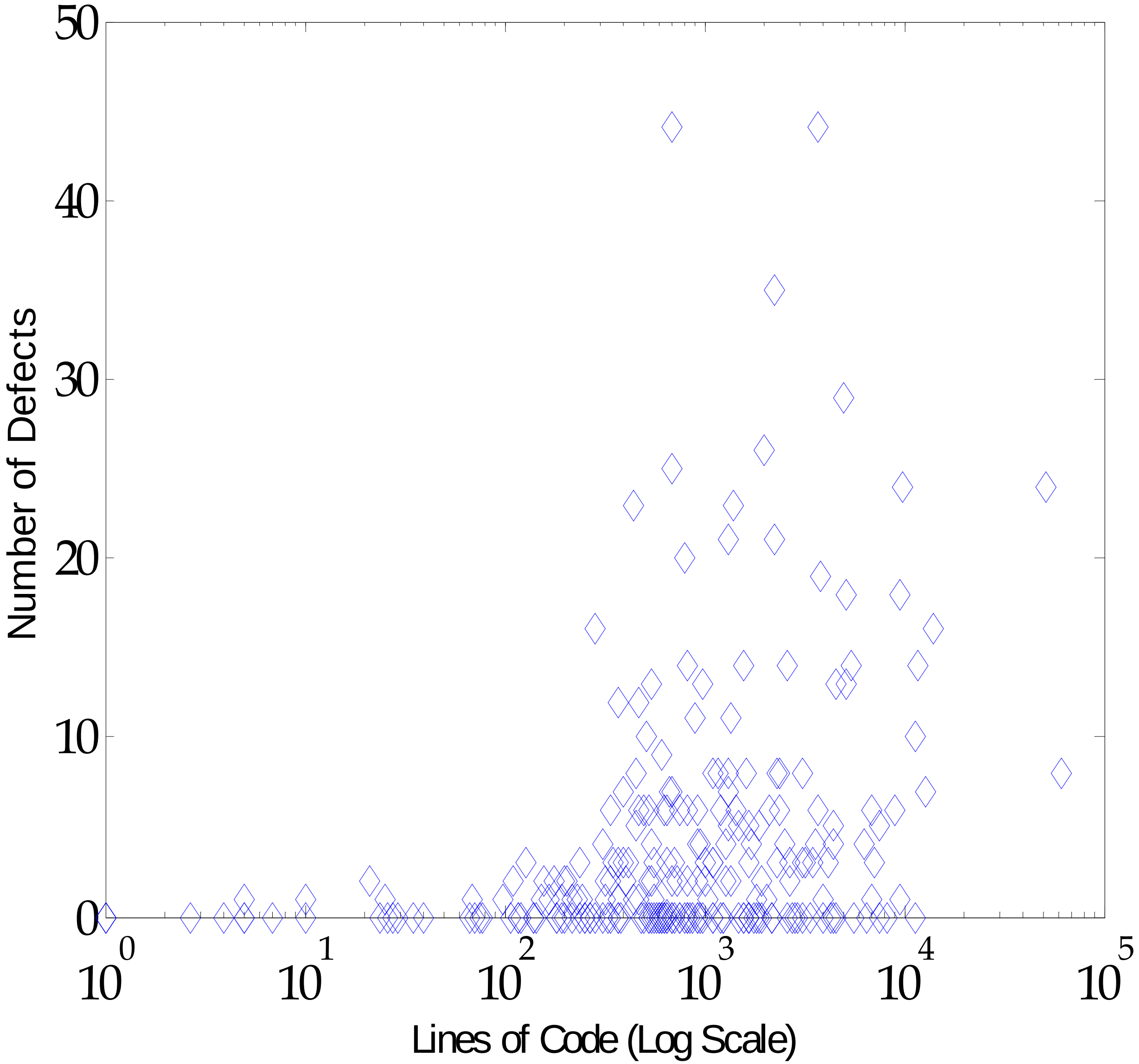
Business
Transformation
Institute

Data Model Assumptions

- We know that $D_{i,j} = d_{i,j} + e_{i,j}$.
- We do not know the form of the functional relationship between the $d_{i,j}$ and $e_{i,j}$ and the properties of the code C_j and the random effects $\varepsilon_{i,j}$.
- We will make the same assumptions for our HMM model that is made in control chart statistical process control:
  - The contribution of the $e_{i,j}$ can be controlled through training and analysis of the effect of different review teams and reviewers.
  - The dominant factor in the code properties is code size and measured by lines of code.
  - The relationship between lines of code and the $d_{i,j}$ is linear.

Business Transformation Institute

Relationship Between Defects and Code Size



Business Transformation Institute

Defects, Code Size Model, and HMMs

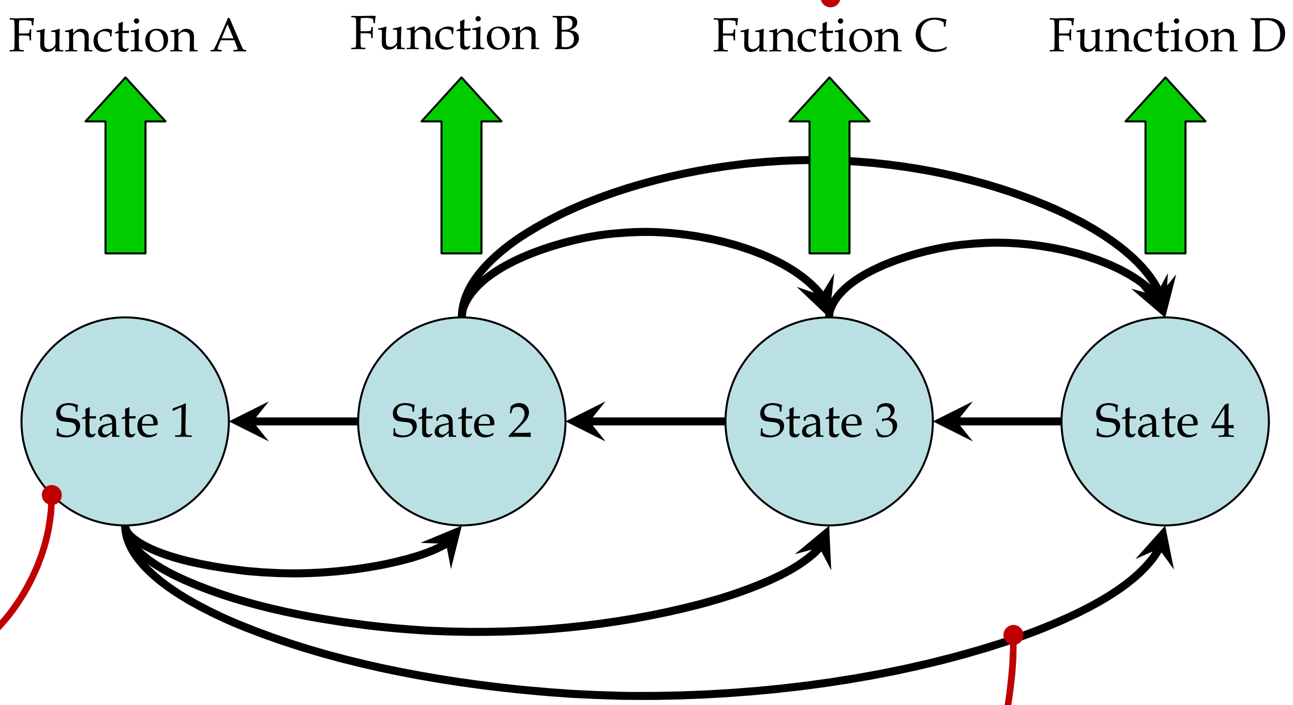
- Using the assumptions, we write: $d_{i,j} = \varepsilon_{i,j} C_j$ and hence $\varepsilon_{i,j} = d_{i,j}/C_j$.
- We will seek to model the $\varepsilon_{i,j}$, across all of the projects, as being drawn from different probability distributions depending on the underlying state of our software development as being “in control” or “not in control”.
- To be able to understand these two states, we need to train our HMM.

Training the HMM

- The steps in training the HMM:
  - Analyze the data to support a reasonable first guess for the underlying Markov state structure and for the HMM parameters.
  - Iterate the guess using the Baum-Welch algorithm to improve our model estimates.
    - The Baum-Welch algorithm is implemented using the Matlab function “HMMTrain”.
- The HMM parameters to be initially estimated are:
  - The number of states in the underlying Markov model.
  - The transition probabilities from any state in the Markov model to any other state.
  - The emission probability distributions that generate the values $\varepsilon_{i,j}$ that we observe.

Training the HMM continued

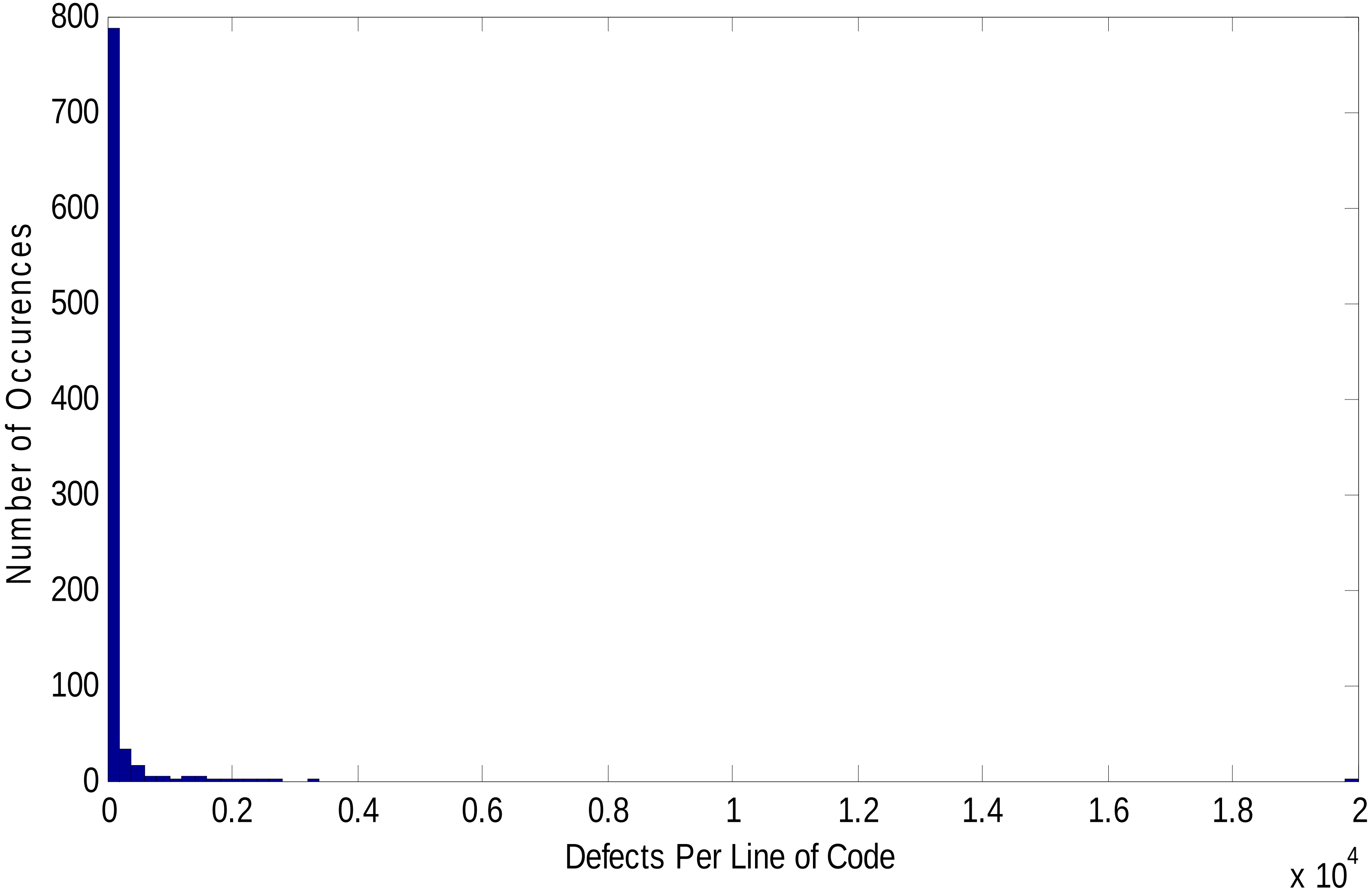
Estimate the output functions



Estimate the number of states

Estimate the transition probabilities

Defect Rate Histogram



Analysis of the $\varepsilon_{i,j}$ Term

- Question: Do the $\varepsilon_{i,j}$ all come from the same distribution regardless of the originating project?
  - Answer: If we test the observed defects $D_{i,j}$ to see if they all come from the same Poisson distribution, the answer is **no (rejected at 99.9999% confidence**).
    - The Poisson distribution is often used for modeling defect generation.
  - Answer: If we use the non-parametric Kruskal-Wallis test, then we again reject that all of the samples come from the same distribution.
  - Answer: If we do pair-wise comparisons using Kruskal-Wallis, then we find that most of the data points are *not* contributing to the dissimilarities in distributions.

Analysis of the $\varepsilon_{i,j}$ Term

- Question: Are any of the distributions Poisson?
  - Answer: If we use the "index of dispersion" to test if the observed defects $D_{i,j}$ for each individual project come from (possibly different) Poisson distributions, the answer is that 19 of the 47 projects might reasonably come from some Poisson distribution.
- Question: Are the Poisson distributions the same?
  - Answer: No. However, further analysis reveals that 9 of the 19 projects appear to have one Poisson distribution and 10 of the 19 some other Poisson distribution.

The Plan for Training the HMM: Estimate for the Number of States

- Treat the underlying Markov model as having three states:
  - These states reflect that the data seems to indicate that there are at least two Poisson distributions and one unknown distribution contributing to the observed values.
  - The interpretation of these states could be that there is:
    - One "in control" state corresponding to one of the apparent Poisson distributions.
    - One "in control but different" state corresponding to the other of the apparent Poisson distributions.
    - One "not in control" state corresponding to the unknown distribution.

Business
Transformation
Institute

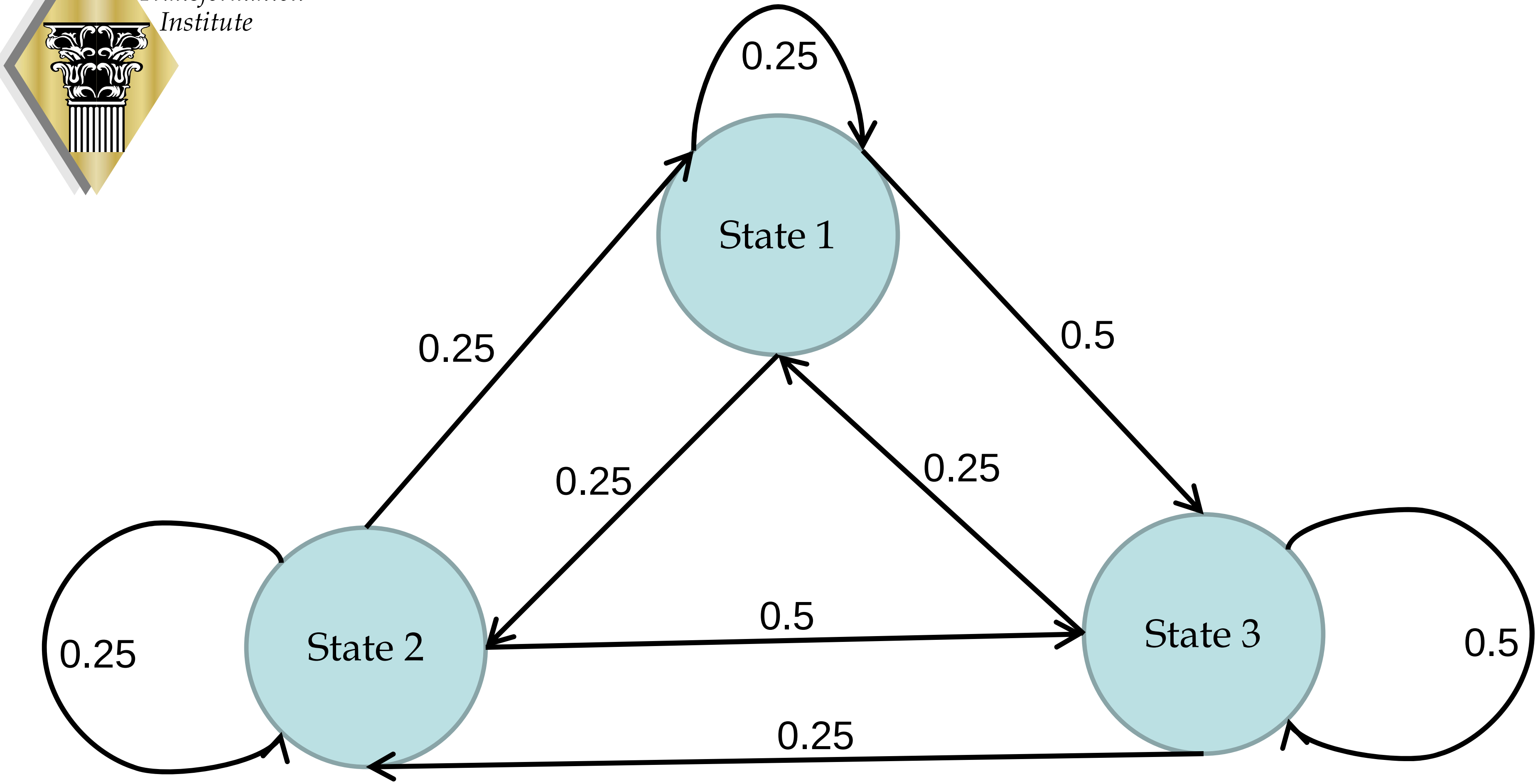
The Plan for Training the HMM: Estimate for the Transition Probabilities

	State 1	State 2	State 3
State 1	0.25	0.25	0.5
State 2	0.25	0.25	0.5
State 3	0.25	0.25	0.5

- Why these values?
  - 19 of the 47 data sets (roughly half) are Poisson—corresponding to states one or two.
  - 9 of the 19 data sets (roughly half, or one-quarter of all of the sets) correspond to one Poisson distribution (state one).
  - 10 of the 19 data set correspond to the other Poisson distribution (state two).
- Remember: the sum of two independent Poisson distributions is itself Poisson.

Business Transformation Institute

Initial Markov Model



The Plan for Training the HMM: Estimate for the Emission Functions (Alternate 1)

- Normalize the $\varepsilon_{i,j}$:
  - Treat the emission functions coming from each Markov state as being from one of three normal distributions .
  - The normal distributions from the "in-control" states will be estimated as having a low mean—an arbitrary value of 0.25. (Variance set to 1.)
  - The normal distribution from the "not in-control" state will have four times the mean for the "in control" states.
  - This normalization estimation is a good way to compare HMM results to control chart results—we're seeing if we can "beat a control chart at its own game".

The Plan for Training the HMM: Estimate for the Emission Functions (Alternate 2)

- Given the observed defect rates $\varepsilon_{i,j}$, scale each defect data point to the number of defects that would have been observed on a standard code set of 1000 lines of code.
  - Treat states one and two as generating Poisson distributions and estimate the mean of each distribution using the sample mean.
  - Treat state three as generating a negative binomial distribution NegBinom(r, p), where $0 < p < 1$ and $r > 0$. We will arbitrarily set r small (r=10) and estimate p.
    - This negative binomial distribution is a "contagious" rare event distribution.
  - This estimate method avoids the "blunt instrument" convergence-in-the-limit style of invoking the central limit theorem as in the first alternate.

Using HMMs as an SPC Tool

- The principal application for HMMs as an SPC tool is to help understand the progression through the state sequence.
  - Once the HMM is trained, we can use the HMM on other data sets to determine the progression from one state to another.
  - Our analysis may equate one or more states with “in control” or “not in control”.
- We will train the HMM using the longest 10 project data sets.
  - Some of these data sets include points that control chart techniques label as “not in control”.

Model Training: Convergence

- An HMM converges when two successive steps are within a small tolerance of each other.
  - Tolerance set to 0.000001.
- Under either emission model assumption, the HMM converges.
- Convergence is much faster under emission alternate 2.
  - Convergence occurred in 39 iterations for emission alternate 2.
  - Convergence occurred in 73 iterations for emission alternate 1.

Model Training:
State Transition Matrix

- The final estimated state transition matrix indicates that transitions to "not in-control" states are far less likely than the initial guess.
- "Not in-control" states are not particularly persistent.

	State 1	**State 2**	**State 3**
State 1	0.83	0.14	0.03
State 2	0.28	0.59	0.13
State 3	0.31	0.37	0.32

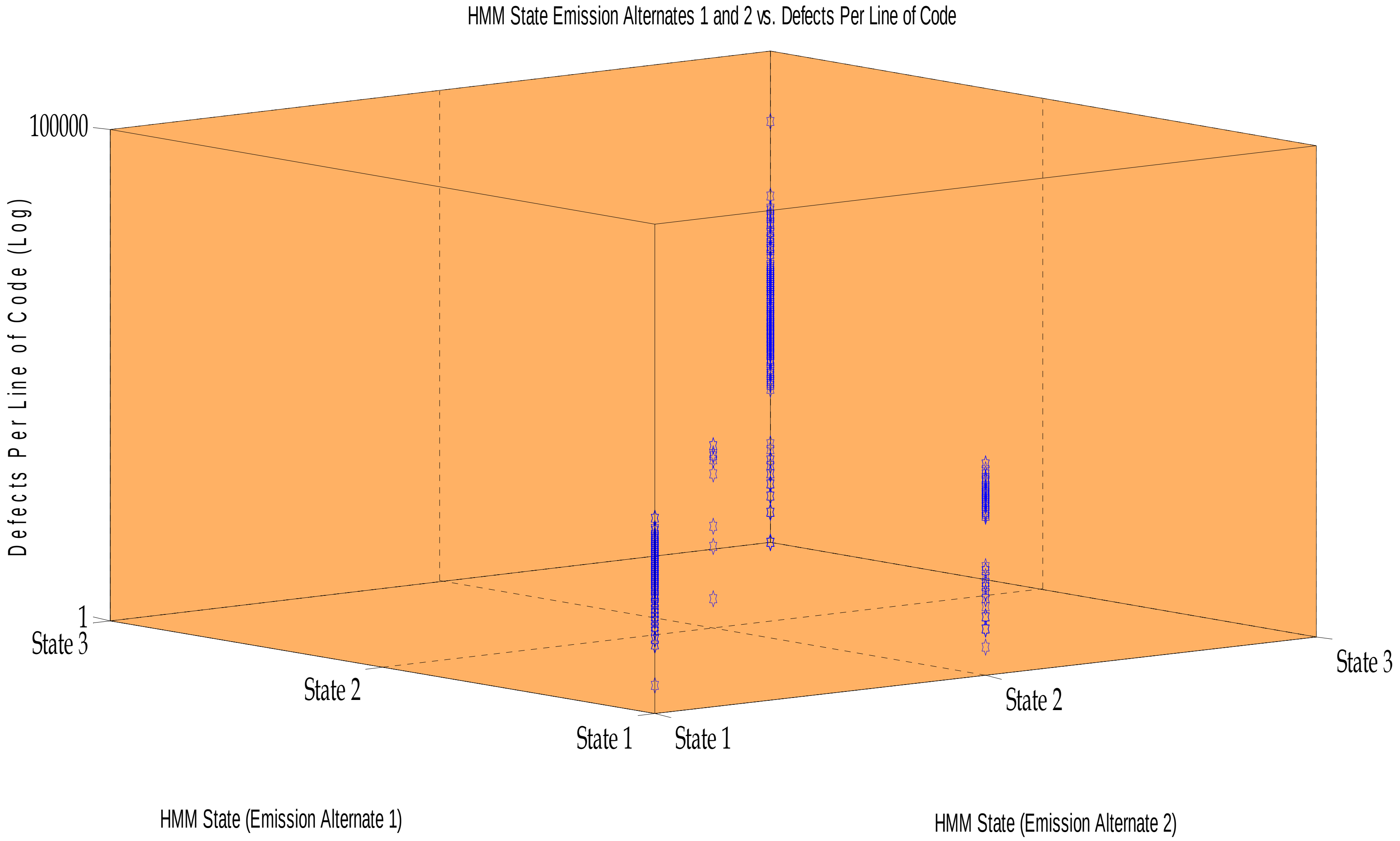
- The existence of state 2 may imply process optimization opportunities.

HMM vs. Control Chart Analysis

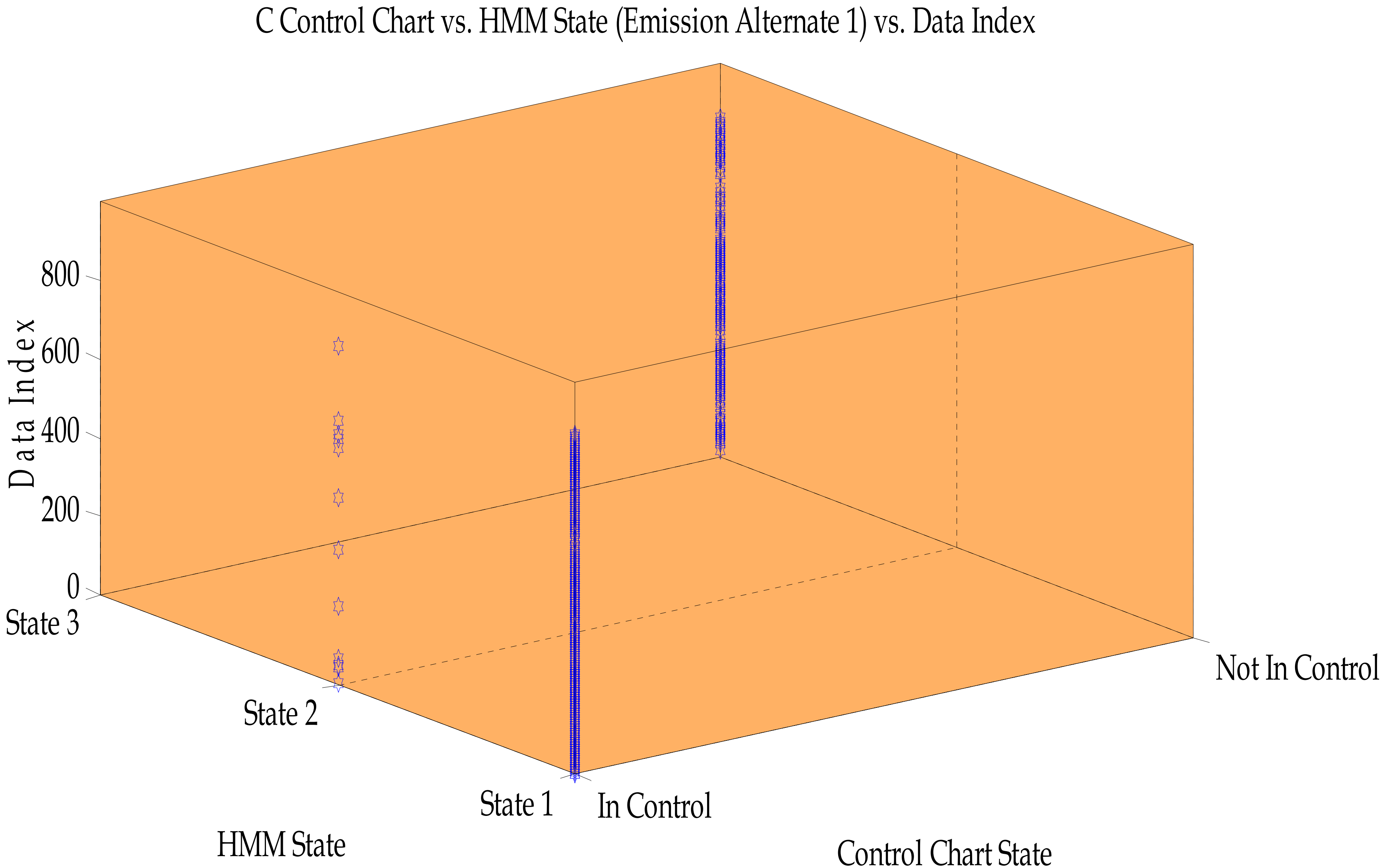
- The following slides compare the analysis of the data points.
  - Emission function alternate 1 vs. 2:
    - State sequence based on emission function 2 is more conservative—more likely to label a point as state 2 or 3.
  - C control chart vs. emission function alternate 1:
    - Both agree on data points that are not in control.
    - Emission function 1 labels some in control points as state 2.
  - C control chart vs. emission function alternate 2:
    - Both agree on data points that are not in control.
    - Emission function 2 labels more in control points as state 2.

Business
Transformation
Institute

Data Point Classification: Emission Alternate One vs. Two

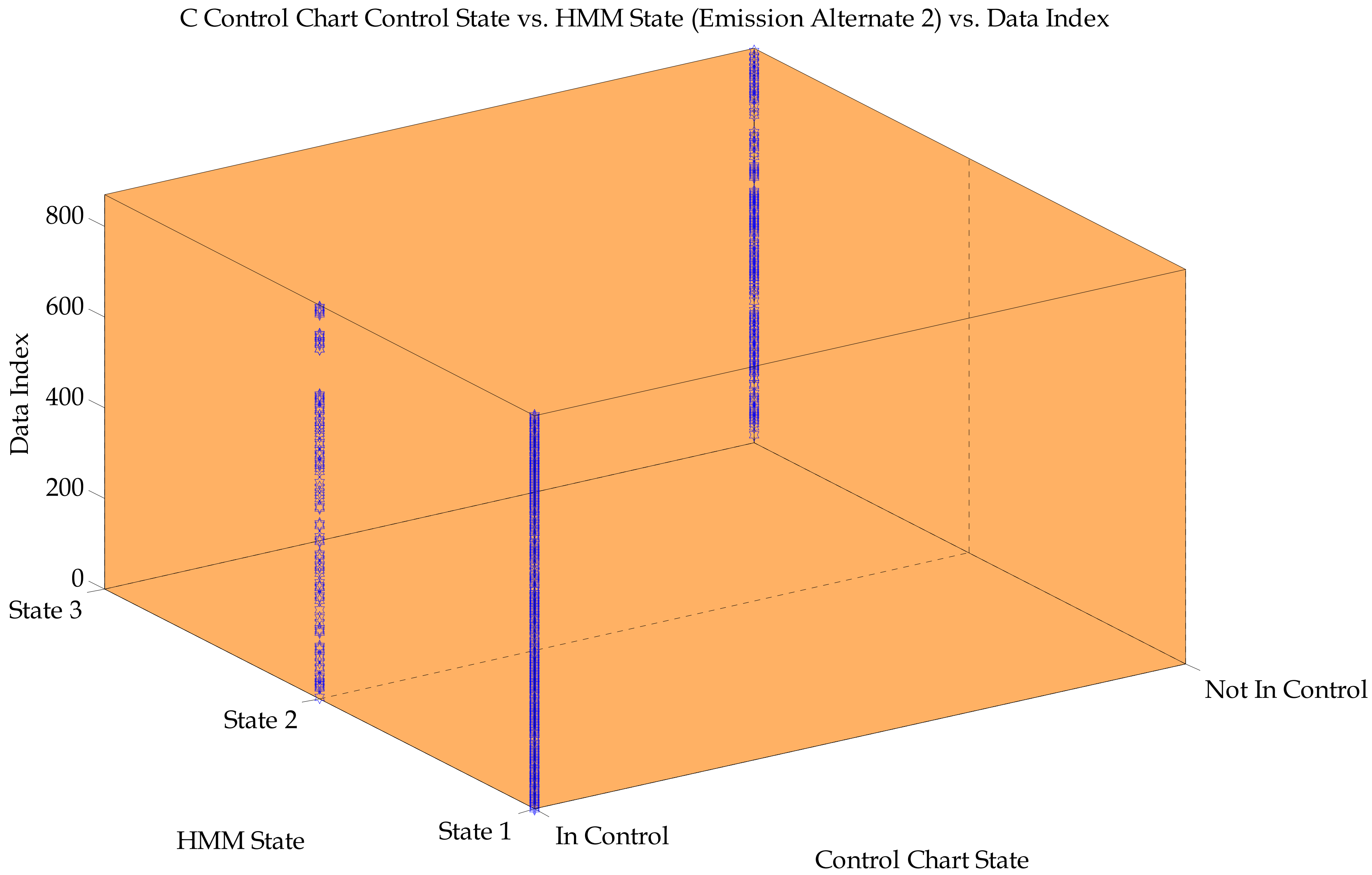


HMM Alternate 1 vs. C Control Chart



Business
Transformation
Institute

HMM Alternate 2 vs. C Control Chart



Conclusions

- HMMs can match or exceed the performance of control chart techniques in identifying not in control states.
  - HMMs may give greater insight in support of process optimization.
- HMMs are not as easy to use as control charts.
  - More analysis of the data is required to build to train the HMM.

Contact Information

Bob Moore
Business Transformation Institute, Inc.
Phone: 410-997-1237
Email: rlmoore@biztransform.net

Ray Luke
Raytheon Space & Airborne Systems
Phone: 972-952-3193
Email: r-luke@raytheon.com

Tony Fields
Raytheon Space & Airborne Systems
Phone: 972-952-4295
Email: tdfields@raytheon.com