Data Learning Based Diagnosis

Li-C. Wang
University of California, Santa Barbara

The high-level picture

- Predictability directly impact yield
- As technology scales, we encounter
  - Lower predictability
    - Result in large margins in design
  - Either we lose yield or lose design resources to have unnecessary margins

- Ultimately, we want to optimize design effort while achieving a desired yield

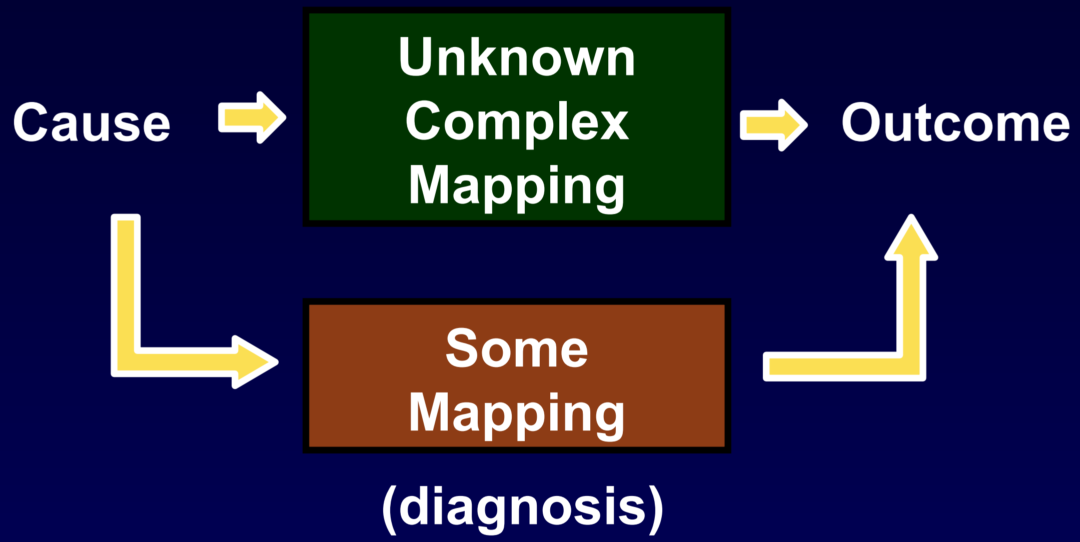
Design for Reality

- Have a tool that we can trust can greatly improve our ability to predict

- Where is the reality?
  - The reality is reflected in TEST DATA

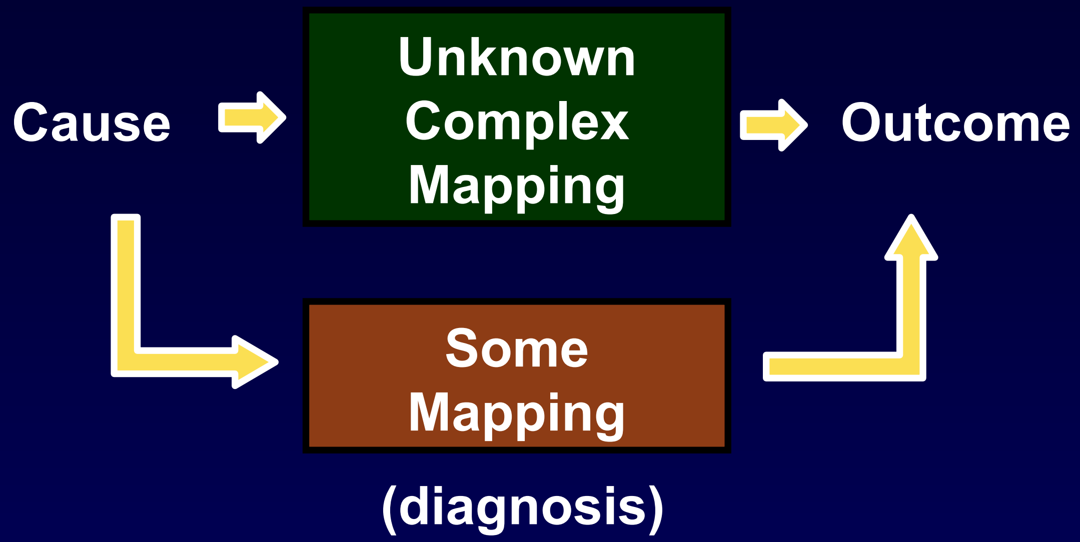
- Hence, the interest lies in “diagnosis”

Diagnosis



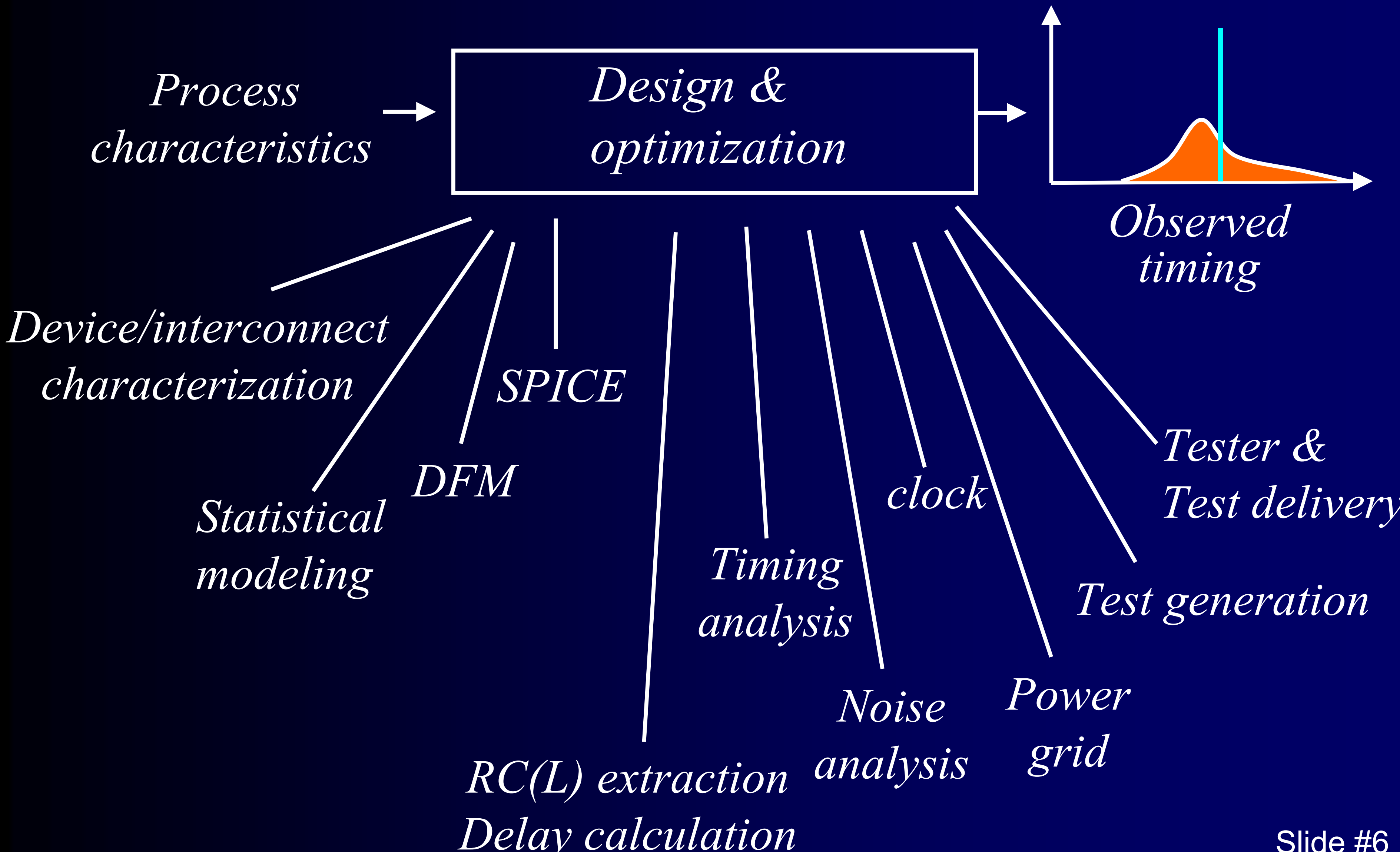
- Diagnosis is to infer “causes” from known “outcome” or “evidences”
- Typically, we establish “some mapping” from cause to outcome in order to evaluate the “fitness” of a cause

Two types of diagnosis

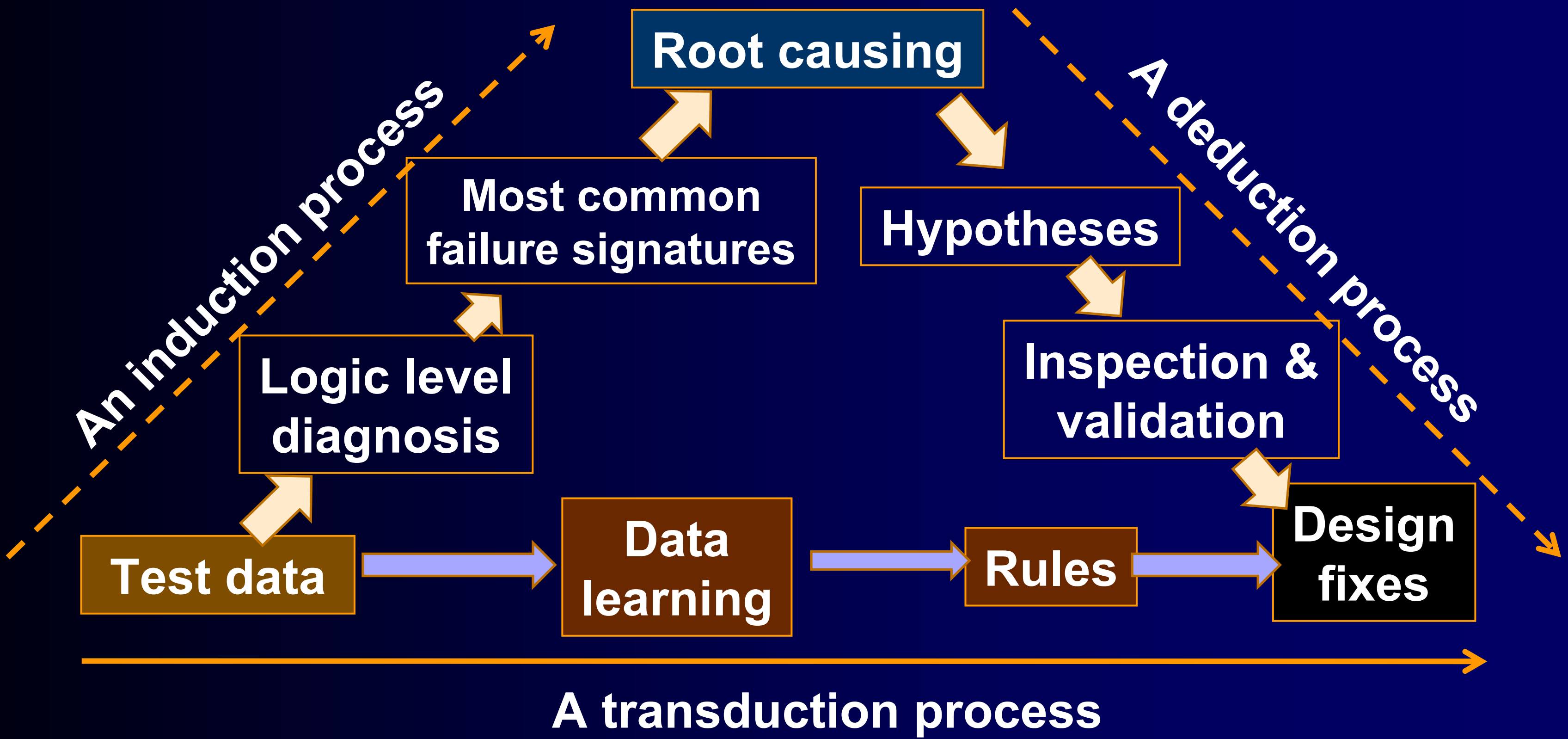


- Fixing a mapping, then find the best cause to explain the outcome under that mapping
  - eg. Traditional, using a fault model
- Without fixing a mapping, find the “simplest” mapping that allows a cause to “almost perfectly” explain the outcome
  - eg. Data learning based diagnosis

Fault model doesn't work!
Low yield can be due to too many reasons



Data learning based diagnosis – Another perspective



- Transduction diagnosis – bypass the most difficulty "root-causing" step
- Get to the fixes directly from data

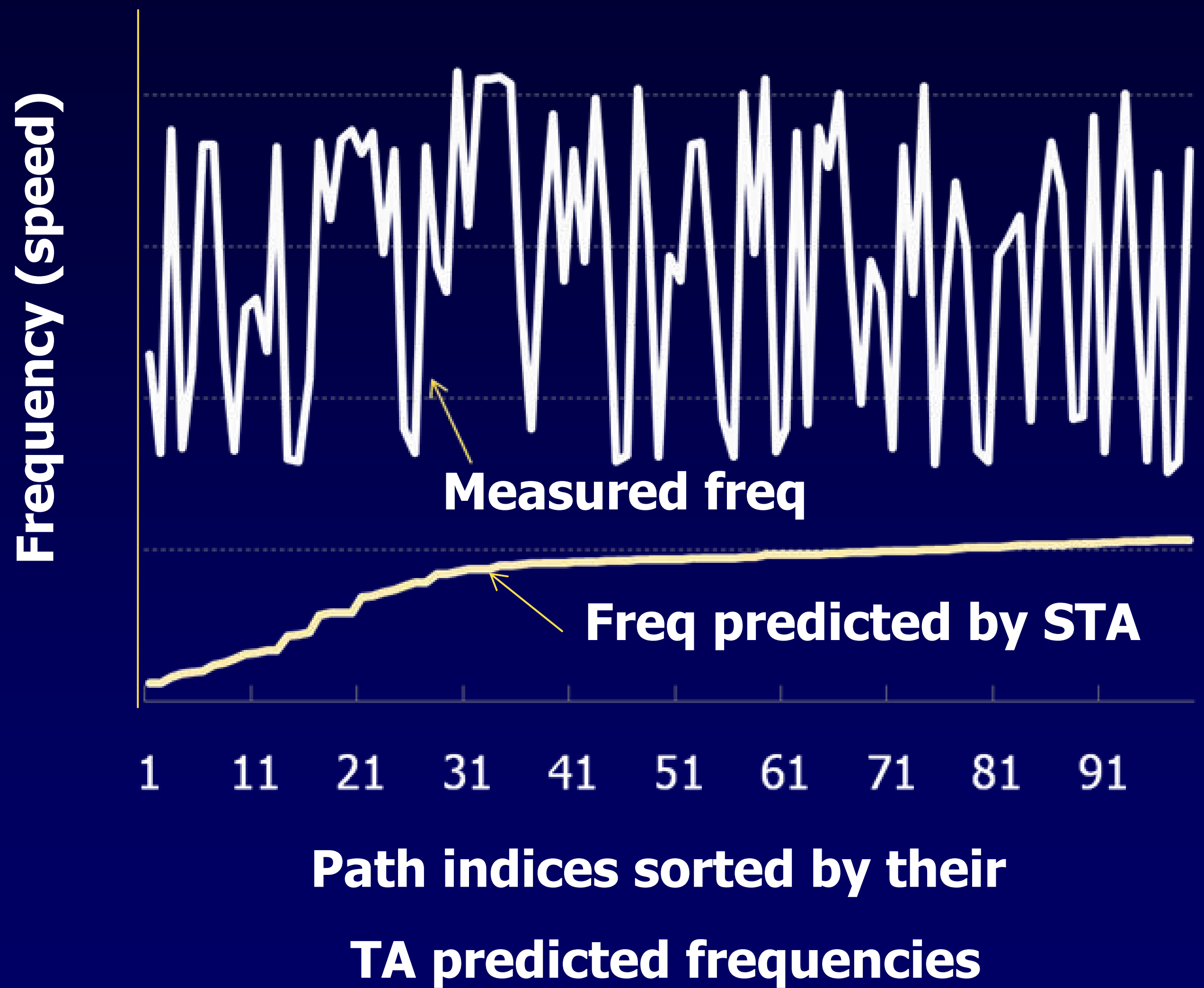
One popular application

- One common application is to calibrate a timing analysis flow
- What kind of test data to use?
  - Let's start with measurement data on paths
- Our objective
  - To find "simple rules" that can improve the predictability of our timing flow

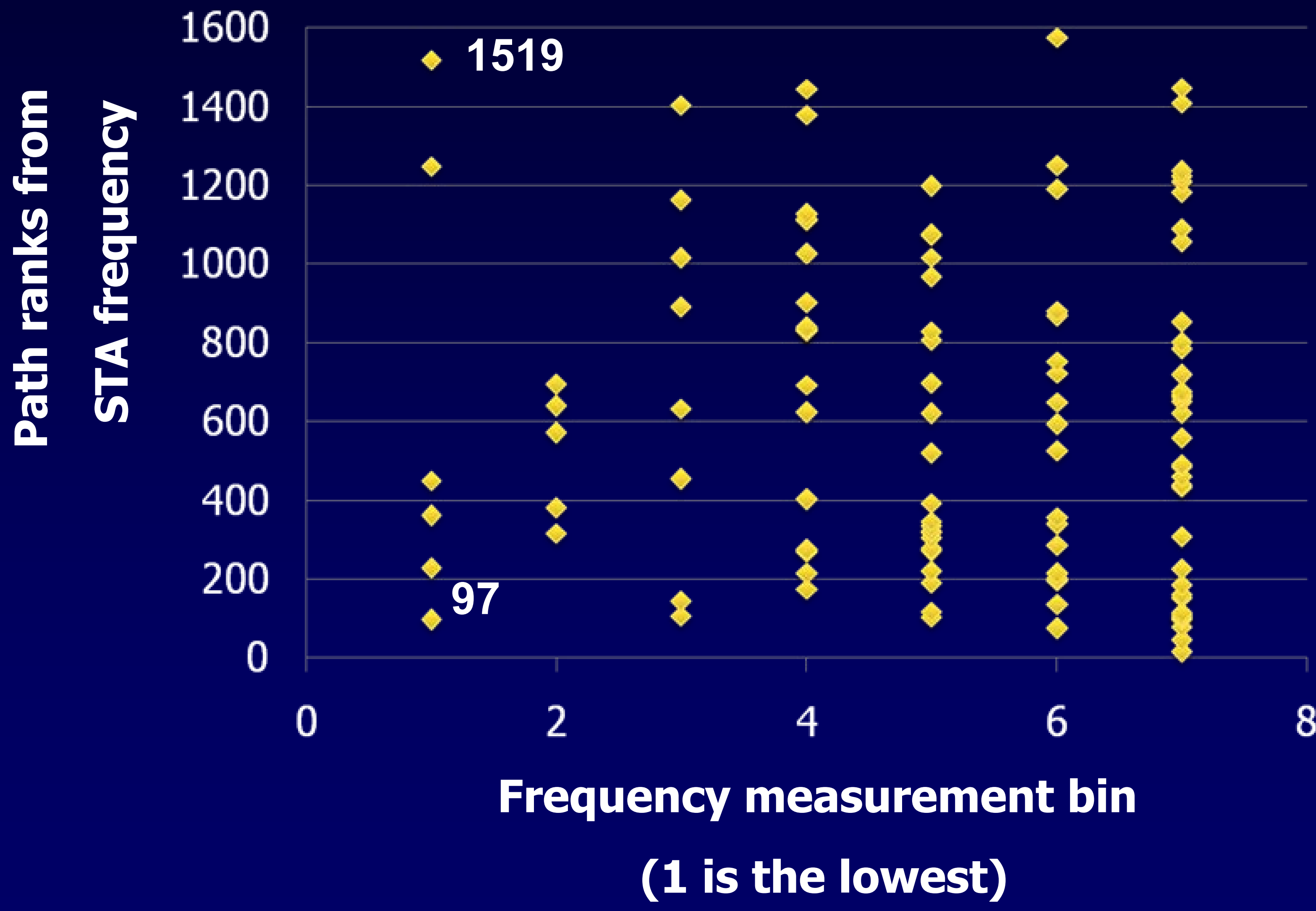
An example chip

- A sub-GHz network processor
  - Manufactured in summer, 2008
- 1622 paths selected from the first 10000 timing critical paths
  - Test pattern for each path
- 1ps frequency stepping to measure the delay of every path

STA vs. 1 silicon chip



Silicon vs. STA



No clear answer!

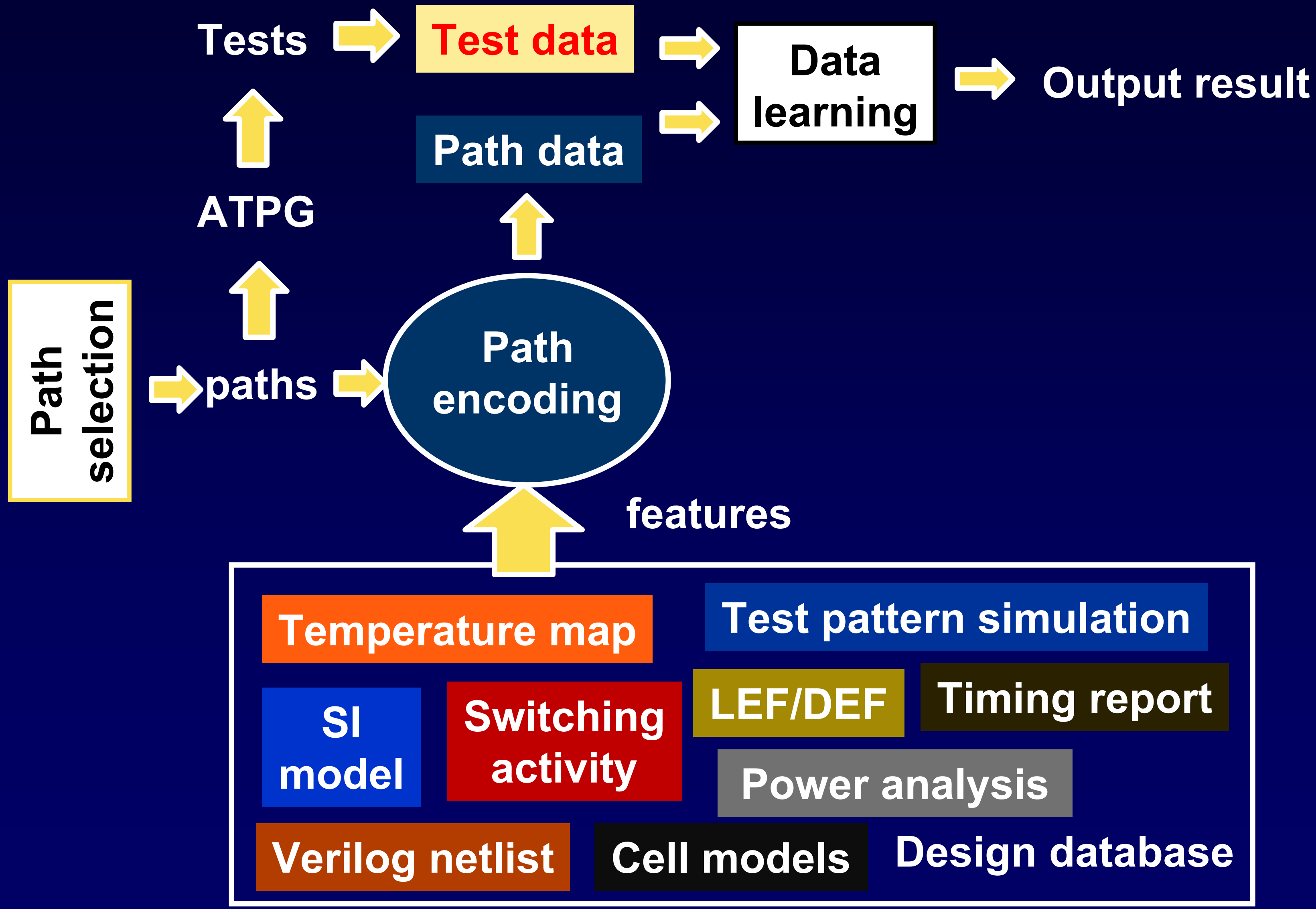
- Depending on who you talk to, you hear various answers for the poor predictability
  - Timing tool is not setup right
  - IR drop causes some paths to have more delay
  - Weak gates are placed along critical paths
  - Lithographic issues
  - Stacked vias
  - Cross coupling noises
  - Temperature issues, ex. inversion behavior
  - Test measurement inaccurate
  - And many more …

- The truth is:
  - No body really know which is more important
  - Everyone is promoting an answer they are familiar with

Use the proposed diagnosis approach

- Use a set of paths to observe design-silicon correlation
  - Less intrusive
  - If you have ROs, they will help
- Extract a set of "path features" from design database to encode path characteristics
  - Potential un/mis-modeled effects
  - Features can be company proprietary information (ex. point to process/design methodology holes)
- Learn from design data and silicon data
  - Formulate it as a data mining problem

Overall diagnosis flow



Two application scenarios

- (1) When you can divide paths into two classes where each class has many paths
  - To explain “group of behavior”
- (2) When you have a few paths that are special and distinguished from the majority of the paths
  - To explain “outliers”

Scenario (1)

- When you have two classes of paths
  - Our goal is to identify the most important few features that differentiate the two classes
- An example
  - Class A has all paths whose measured delays are greater than predicted delays
  - Class B has all other paths
  - The selected features tell what important factors are to cause the under-estimation
- Another example
  - Class A has all paths whose measured delays are much different from predicted delays
  - Class B has all other paths
  - The selected features tell what important factors are to cause the mis-prediction (either under- or over-estimation)
- One can devise the scenario to for a particular question of interest

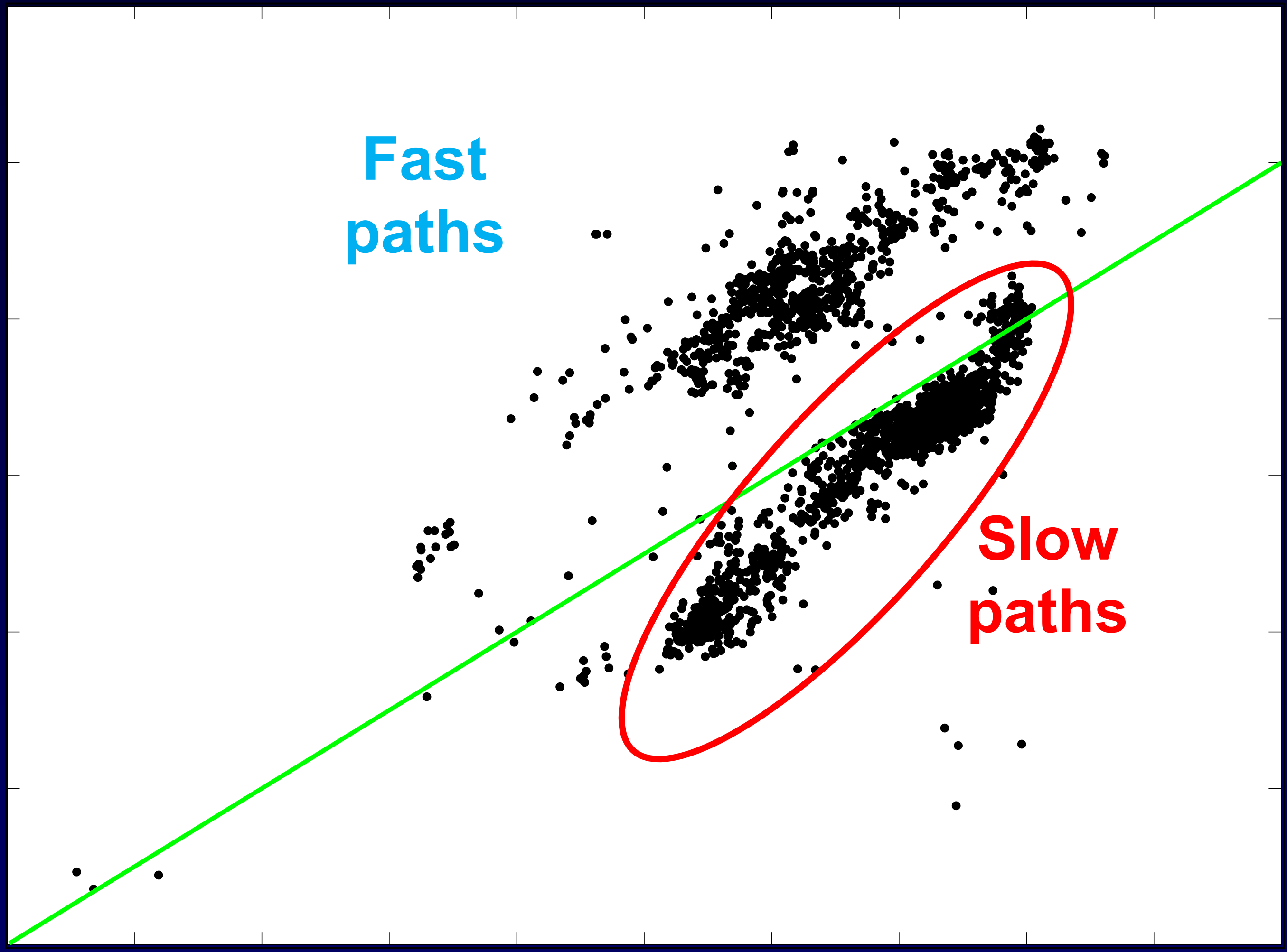
Scenario (2)

- When you have a few special paths
  - Our goal is to explain why those paths are special
- An example
  - The special paths can be the most critical paths seen in test measurements but not properly predicted by STA
  - We will find a combination of features to form “rule” to explain the reason why those paths are special
- Another example
  - The special paths are “high-variability” paths, i.e. measured delay variance is bigger across multiple dies
  - The rule extracted can be used as a design rule to avoid high variability
- Again, one can devise the scenario to for a particular question of interest

On-going experiments

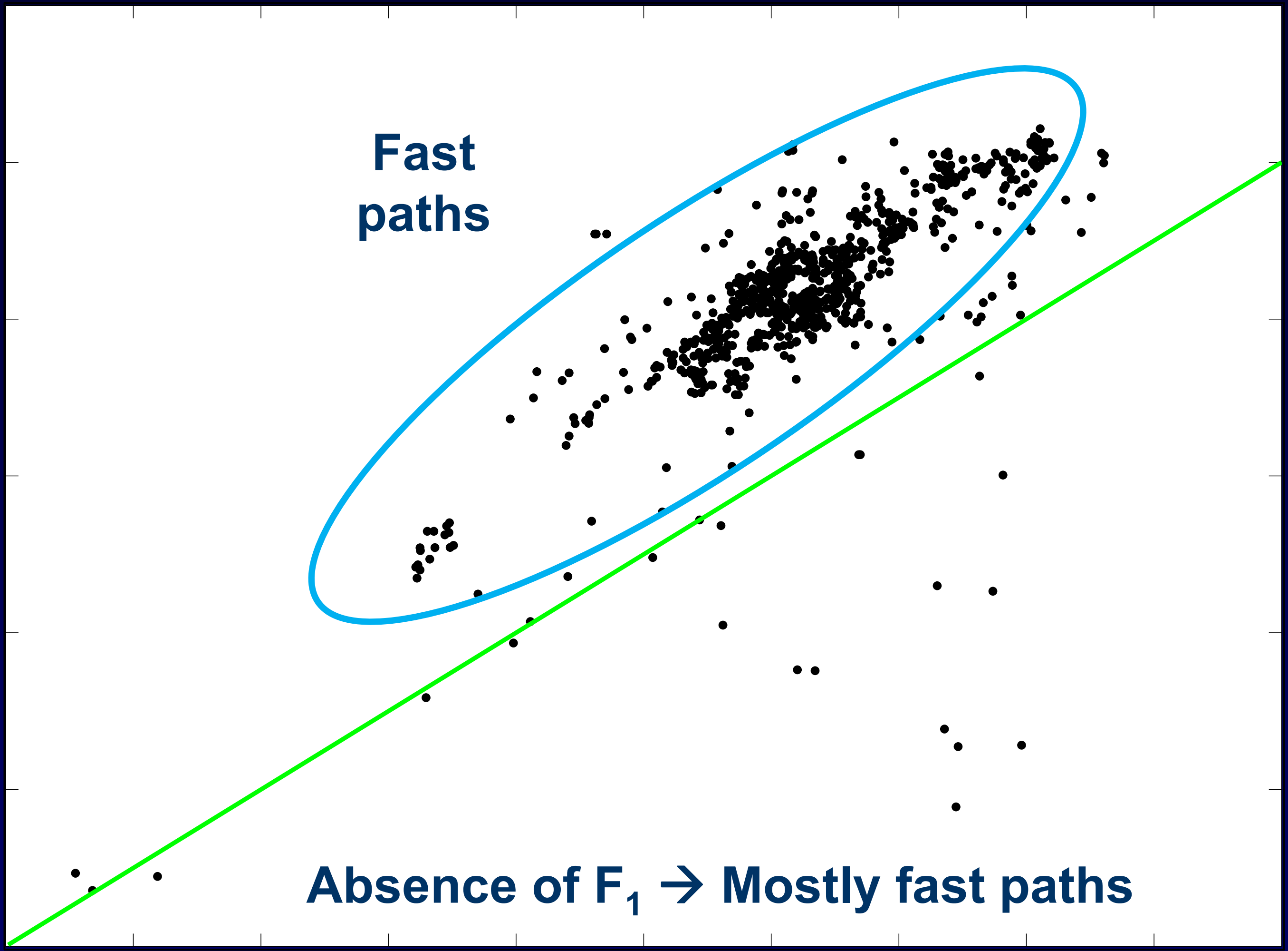
- We collect data on 30 dies
  - Measure delays on 2143 paths
- We collect all design data for the 2143 paths
  - Derive various features from layout and logic structure
- We tried to answer several questions
  - Why some paths are mis-predicted long paths?
  - Why some paths have high variability?
  - Why two clusters of paths? (see next slide)?

Roughly two clusters



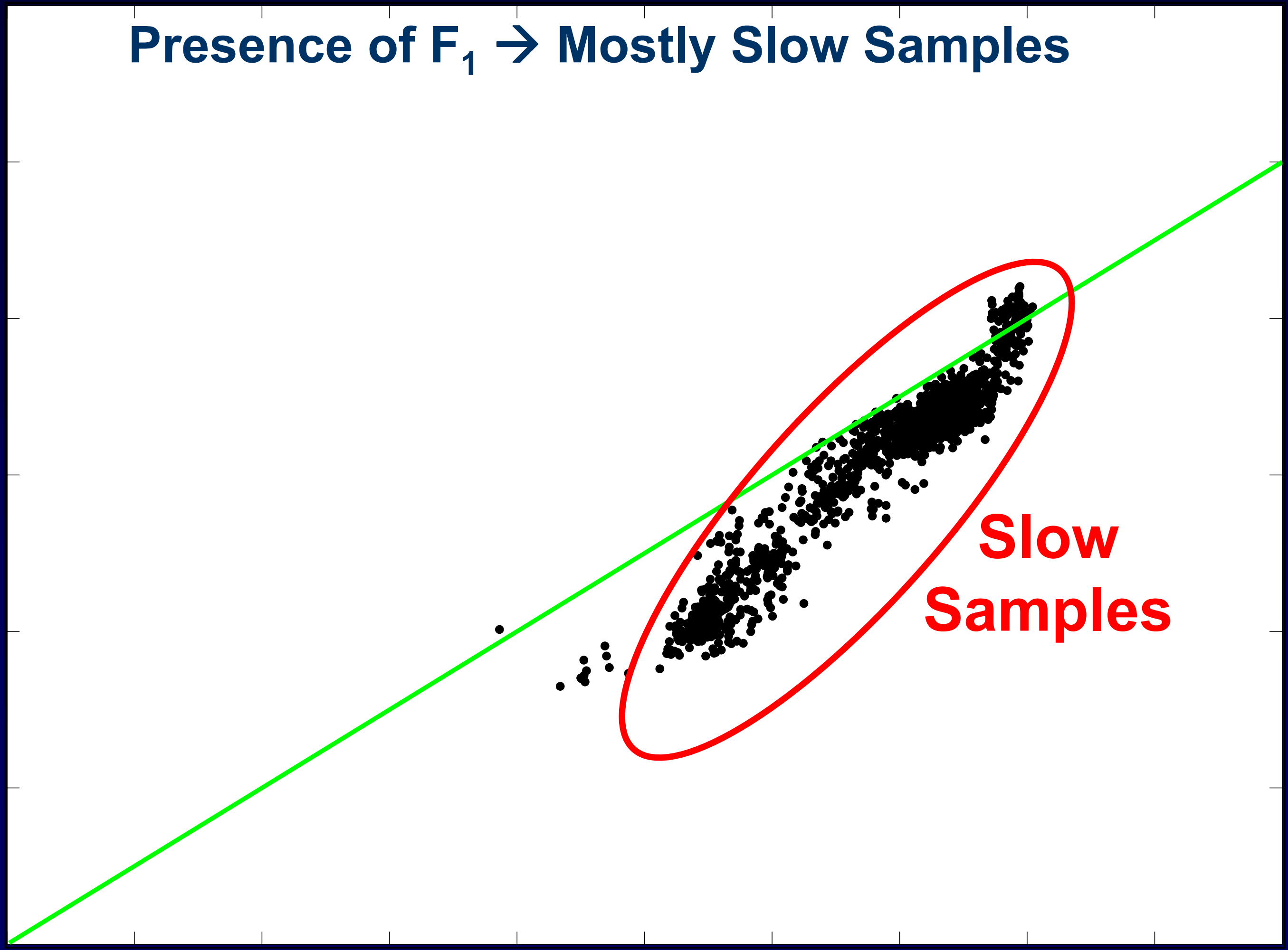
- Each delay is average over 30 dies

Absence of a casue



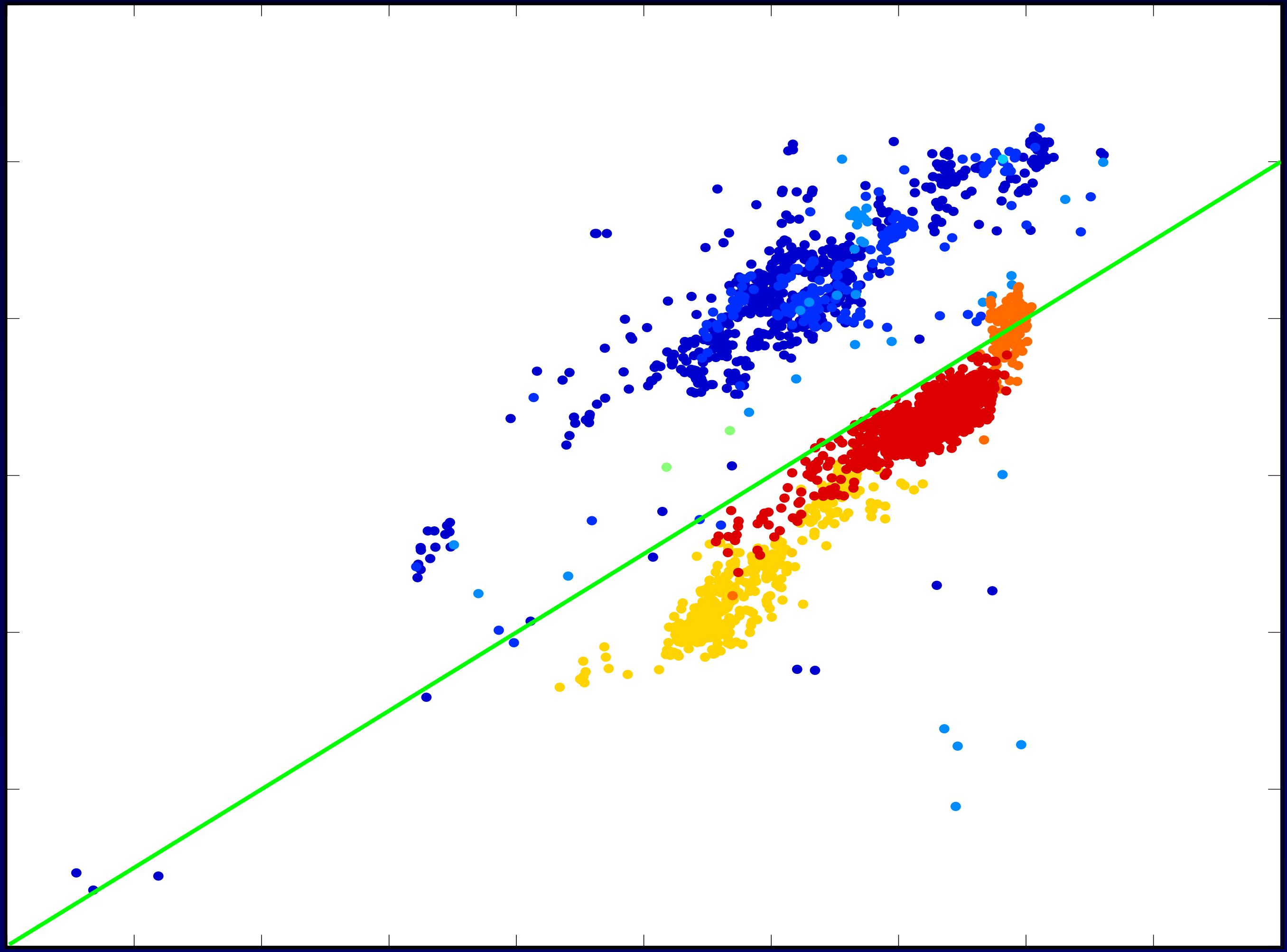
- Each delay is average over 30 dies

Presence of the cause



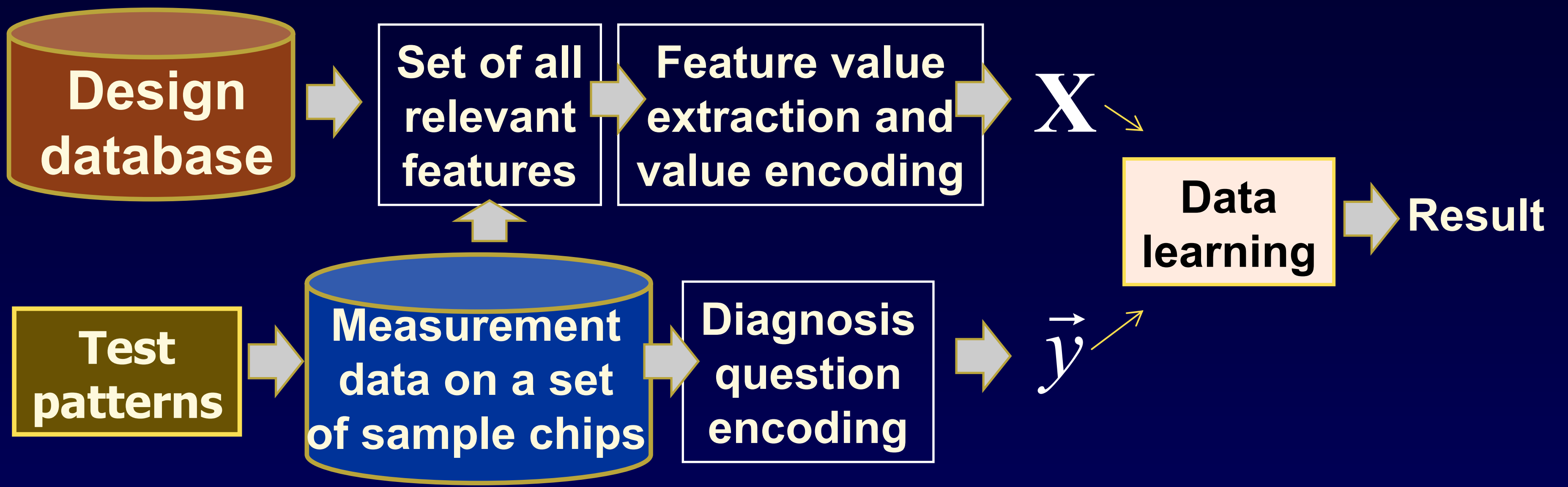
- Each delay is average over 30 dies

Combined causes



- Can explain the data well with combined causes

Summary of the diagnosis flow



- On a given test dataset,
  - (1) Diagnosis question encoding
  - (2) Extract all relevant features
  - (3) Encode feature values
  - (4) Apply suitable data learning methods
- Validate results (with designer, etc.)

Other application scenarios

- The proposed framework has been applied to other scenarios
  - Analyze why a few paths become silicon “speedpaths”
  - Analyze layout features corresponding to common logic failures
  - Analyze customer returns and why they passed the ATE test
- Collaborative companies: AMD, Freescale, Intel

Conclusion

- Design for Reality – Improving predictability
- Traditional fault model diagnosis doesn't work for this type of application
  - We need a data learning approach
- The proposed diagnosis approach can be applied in diverse scenarios
- We are in the process of expanding "data mining" to "knowledge discovery"
  - Meaning that we want to uncover rules that are not only "statistically significant rule" but also "intuitively explainable" based on design knowledge