Verification and Validation

Overview

- It is very simple to create a simulation!
- It is very difficult *to model something accurately*.
- In this lecture we will investigate:
  - ideas of model verification,
  - validation,
  - and credible models.

Verification and validation (V&v)

- V&v is the process of checking that a product, service, or system meets specifications and that it fulfills its intended purpose. These are critical components of a quality management system such as ISO 9000. Sometimes preceded with "Independent" (or IV&V) to ensure the validation is performed by a disinterested third party.
- In software project management, software testing, and software engineering, V&v is the process of checking that a software system meets specifications and that it fulfils its intended purpose. It is normally part of the software testing process of a project.

Verification and validation - Definitions

- Also known as software quality control.
- Validation checks that the product design satisfies or fits the intended usage (high-level checking) — i.e., you built the right product. This is done through dynamic testing and other forms of review.
- According to the Capability Maturity Model (CMMI-SW v1.1),
- Verification: The process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. [IEEE-STD-610].

1. *Department of Defense Documentation of Verification, Validation & Accreditation (VV&A) for Models and Simulations*, Missile Defense Agency, 2008

… Verification and validation - Definitions

- Validation: The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements. [IEEE-STD-610]

Within the modeling and simulation community, the definitions of validation, verification and accreditation are similar:

- Validation is the process of determining the degree to which a model, simulation, or federation of models and simulations, and their associated data are accurate representations of the real world from the perspective of the intended use(s).[1]
- Accreditation is the formal certification that a model or simulation is acceptable to be used for a specific purpose.[1]

… Verification and validation - Definitions

- Verification is the process of determining that a computer model, simulation, or federation of models and simulations implementations and their associated data accurately represents the developer's conceptual description and specifications.[1]
- In other words, validation ensures that the product actually meets the user's needs, and that the specifications were correct in the first place, while verification is ensuring that the product has been built according to the requirements and design specifications. Validation ensures that ‘you built the right thing’. Verification ensures that ‘you built it right’. Validation confirms that the product, as provided, will fulfill its intended use.

V & v Classification of methods

- In mission-critical systems where flawless performance is absolutely necessary, formal methods can be used to ensure the correct operation of a system. However, often for non-mission-critical systems, formal methods prove to be very costly and an alternative method of V&V must be sought out. In this case, syntactic methods are often used.
- A test case is a tool used in the process. Test cases are prepared for verification: to determine if the process that was followed to develop the final product is right. Test case are executed for validation: if the product is built according to the requirements of the user. Other methods, such as reviews, are used when used early in the Software Development Life Cycle provide for validation.

Software verification

Software verification is a broader and more complex discipline of software engineering whose goal is to assure that software fully satisfies all the expected requirements.

There are two fundamental approaches to verification:

- *Dynamic verification*, also known as *Test* or Experimentation - This is good for finding bugs. Dynamic verification is performed during the execution of software, and dynamically checks its behavior; it is commonly known as the Test phase. Verification is a Review Process.
- *Static verification*, also known as Analysis - This is useful for proving correctness of a program although it may result in false positives.

Software verification - *Dynamic verification* (*Test*)

- Depending on the scope of tests, we can categorize them in three families:
- *Test in the small*: a test that checks a single function or class (Unit test)
- *Test in the large*: a test that checks a group of classes, such as
  - Module test (a single module), Integration test (more than one module), and System test (the entire system)
- *Acceptance test*: a formal test defined to check acceptance criteria for a software
  - Functional test, and Non functional test (performance, stress test)
- Software *verification* is often confused with software *validation*. The difference between *verification* and *validation*:
- Software *verification* asks the question, "Are we building the product right?"; that is, does the software conform to its specification.
- Software *validation* asks the question, "Are we building the right product?"; that is, is the software doing what the user really requires.
- The aim of software verification is to find the errors introduced by an activity, i.e. check if the product of the activity is as correct as it was at the beginning of the activity.

Software verification - *Static verification* (*Analysis*)

Static verification is the process of checking that software meets requirements by doing a physical inspection of it.

For example:

- *Code conventions verification*
- *Bad practices (anti-pattern) detection*
- Software metrics calculation
- Formal verification

Validation

Validation is the process of checking if something satisfies a certain criterion. Examples would include checking if a statement is true (validity), if an appliance works as intended, if a computer system is secure, or if computer data are compliant with an open standard. *Validation* implies one is able to document that a solution or process is correct or is suited for its intended use.

In computer terminology, validation refers to the process of data validation, ensuring that data inserted into an application satisfies pre-determined formats or complies with stated length and character requirements and other defined input criteria. It may also ensure that only data that is either true or real can be entered into a database.

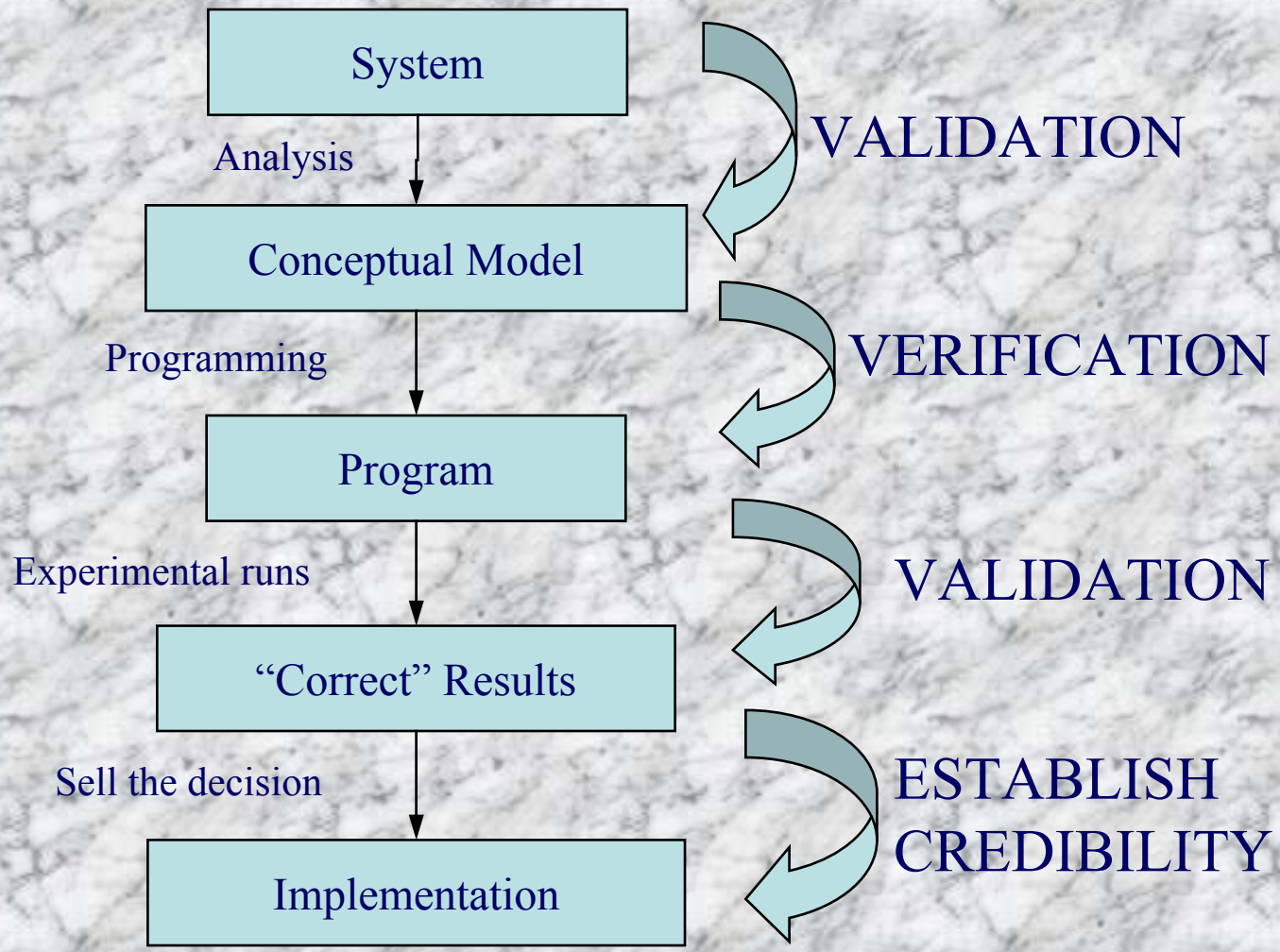
… Validation

- In computer security, validation also refers to the process of assuring or authorizing that a user or computer program is allowed to do something. One method is to use programs such as Validate (McAfee) to check program and data checksum values.
- In the computer architecture and hardware world, validation refers to the process of verifying that the operations of the piece of hardware or architecture meets the specification. In some cases, validation not only refers to finding bugs in the hardware but also proving absence of certain critical bugs which may not have workarounds and may lead to project cancellation or product recall.

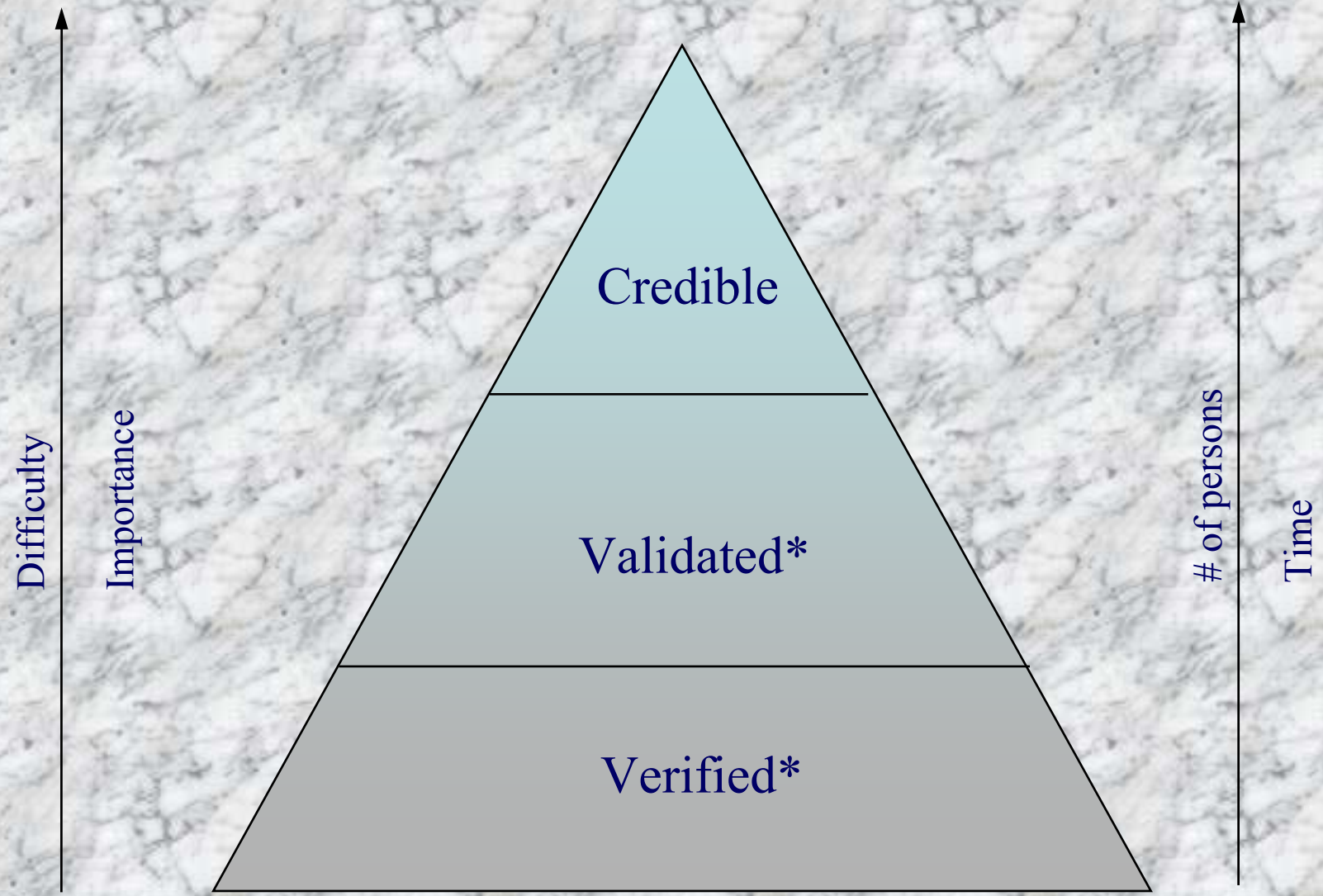
Definitions in *Mod&Sim*

- ***Verification*:** The process of determining that the computerized representation of our system functions as intended.
- ***Validation*:** The process of determining that the whether our model accurately represents the system under study.
- ***Credible*:** The process of ensuring that decision makers believe in the results of your model.

System View


System
Analysis
Conceptual Model
Programming
Program
Experimental runs
"Correct" Results
Sell the decision
Implementation
VALIDATION
VERIFICATION
VALIDATION
ESTABLISH
CREDIBILITY

In a Picture



*Necessary, but not sufficient conditions

Verification, Validation & Credibility

Is the PROGRAM correct?

Is the program a correct MODEL?

Is the model correct with respect to the QUESTIONS or DECISIONS under investigation?

Are the decisions ROBUST?

What is the decision's SENSITIVITY to the parameters?

Perspectives on Validation

- Our major contribution is often a model.
- We learn a great deal about building models in school, but very little about ensuring the correctness of our models.
- The literature on validation is quite poor.
- No definite process, measure, test, or method exist that will ensure model validity.

A Diatribe on Validation

- Validation is very important.
- Students (and frequently practitioners) ignore model validity.
- This is the GIGO syndrome: Garbage In - Gospel Out.
- If you learn nothing else in this course, learn how to validate a simulation model.

An acronym for 'Garbage In, Garbage Out', which emphasizes that the output of a system or ANALYSIS is directly dependent upon the quality of the inputs to that system or analysis.

Combating the Garbage-In, Gospel-Out Syndrome, Michael R. Ault

About Validation

- Validation is incorrectly treated as a distinct activity undertaken at the end of a project.
- Validation is a process.
- Validation should be started at the beginning of a project.
- Validation requires the input of many people.
- Validation is an exercise in human relations as well as a technical endeavor.

Validation Literature

- There is a paucity of research on validation.

  (Finlay & Wilson, 1990. Orders of Validation in Mathematical Modelling. *JORS*, 41(2): 103-109)

- No formal method can be applied in all cases and no absolute measure exists for complex models.

  (Law & Kelton, *Simulation Modeling & Analysis*, 1991)

- The function of models is to influence decision makers. Thus acceptance by decision-makers may constitute *de facto* validation.

  (Butler, 1995. Management Science/Operations Research Projects in Health Care: The Administrator's Perspective. *Health Care Management Review*, 20(1): 19-25.**)**

- Some of the better literature talks about validation as being a process.
- Ignazio and Cavalier suggest validation is a process of interacting with decision makers to build their confidence in model results.

  (Ignizio and Cavalier, *Linear Programming*, 1994)

Schellenberger Framework*

Validity has three dimensions:

1. Technical validity: Comparison against a reasonable set of criteria.
2. Operational validity: A subjective assessment of the behaviour of the model.
3. Dynamic validity: The utility of a model over an extended period of time.

* Schellenberger, R.E., (1974). Criteria for Assessing Model Validity for Managerial Purposes. *Decision Science* 5(5): 644-653.

Technical Validity

- **Model Validity:** The degree to which the underlying conceptual model of a system represents reality.
  - List and vet mathematical, content, and causal assumptions.
- **Data Validity:** The degree to which the data used in an instance of decision making is representative of reality.
  - Accuracy, impartiality, and representativeness of the data.
  - The accuracy of the process of data collection and aggregation.
- **Logical Validity:** Describes the fidelity with which the conceptual model is translated to computer code.
- **Predictive Validity:** The ability of the model to produce results that conform to expected output.

Operational Validity

- **Degree of Improvement:** The robustness of the model results as suggested by the degree of improvement.
  - If the model suggests a 60% improvement in performance for a particular option, the impact of error is likely to be insignificant.
- **Model Sensitivity:** The effect of small change in data parameters on model stability.
- **Implementability:** The ability of the model to produce results that can be adopted in practice.

Dynamic Validity

- **Maintainability:** The ease with which the model can be changed over time.
- **Review Process:** The accuracy and completeness of the process of periodically reviewing the model to ensure continues to conform to reality.
- **Update Process:** The accuracy and completeness of the process to periodically update model parameters.

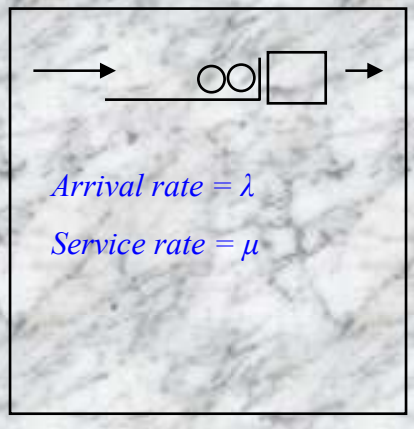
Law and Kelton Framework

- **Verification of Underlying Conceptual Model**
  - Model assumption lists.
  - “Structured walk through” of model with system experts.
- **Verification of Operational Implementation**
  - Structured walk-through of code with experts.
  - Tests under simplified conditions
- **Validation of Operational Model**
  - Vet system results with experts (Turing test)
  - Compare model results against system results.
  - Compare model results against theory or existing models.
  - Empirical tests of model assumptions (sensitivity analysis).

A Plan for Validation – Technical …

- Model Validity:
  - Develop a conceptual model of the system under study.
  - Clearly define (and document) system boundaries, system elements.
  - Develop and document a list of assumptions.
  - Vet the model and assumptions with content experts.
  - Vet the model and assumptions with technical experts.

Conceptual Model



Boundaries

We will include the server and the queue in our model.
We will exclude upstream and downstream processes.

Assumptions

Poisson arrival process.
Exponential service time.

Infinite calling population

FIFO logic…

Review with experts

… A Plan for Validation – Technical …

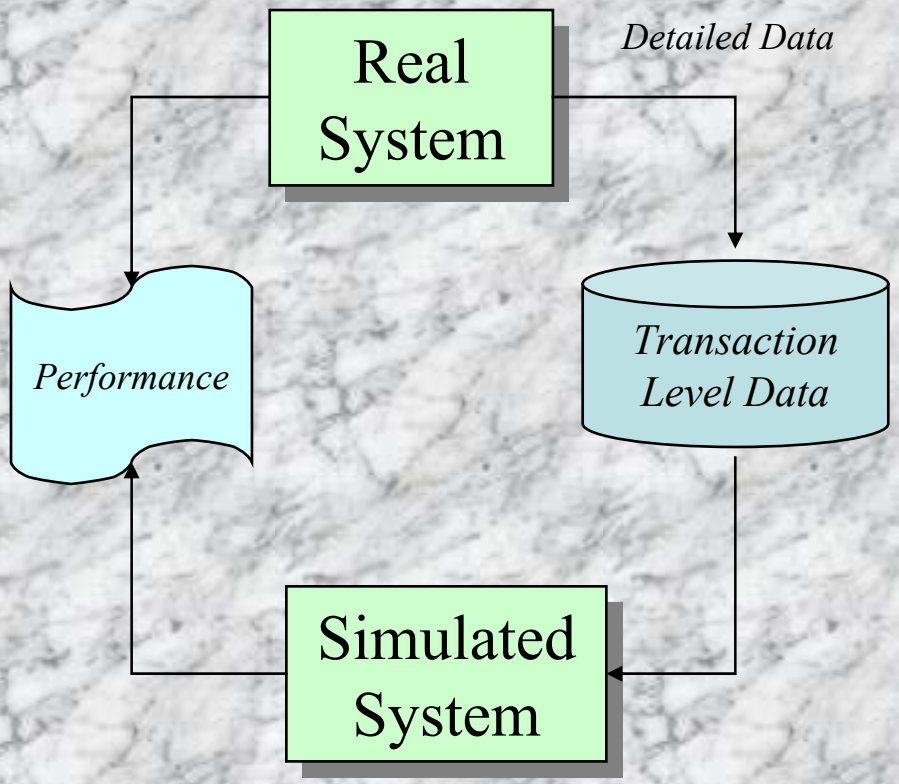
- **Data Validity**:
  - Identify and document data necessary to complete study
  - Identify data sources
  - Carry out preliminary data collection & analysis. Identify and correct any problems in data collection/aggregation
  - Conduct basic tests for data reasonability
  - Develop a statement of data validity

Data Requirements	*Data Sources*	*Preliminary Analysis*	*Data Validity Statement*
Collect time between customer arrivals. *Record service times*	*Hospital ADT system.* *MED2020 patient record system*	*Can we find patient records in both the ADT and MED2020 systems?* *How many records are incomplete?* *What are the max/min times?*	*We feel that the input data is correct because:* *We were able to achieve total compliance with respect to bed days for the months of Jan-Mar*

… A Plan for Validation – Technical

- Logical Validity:
  - Make use of software development tools:
    - Compiler, debugger, and system trace elements.
  - Be a skeptic:
    - Assume that the model is incorrect.
    - Build in elements to test model correctness.
  - Use an evolutionary approach to model development:
    - Build small segments. Test and validate each segment.
    - Take advantage of the sub-network concept in AweSim
  - Make use of experts:
    - Use content experts to review model results (try an animation) throughout the development cycle.
    - Conduct a structured walk through with technical experts.

Software Tools - Trace



- Traces can be powerful debugging tools.
- If possible, take a sample of data from the real system.
- Force your simulation to use the data from the real system line by line.
- If done properly, the two should produce identical results.

A Plan for Validation - Technical

- Predictive Validity:
  - Compare the simulation output against theory (perhaps using simple measures) for which performance is known.
  - Compare the simulation output against existing system output.
    - Inspection (basic or correlated).
    - Confidence interval.

A Plan for Validation - Technical

- Inspection: Compare output of simulation against output of actual system.
  - Basic Inspection: Take a particular performance measure from the simulation and compare it to the output of the real world system.

    $TPUT_S$: 27.8 $TPUT_R$: 29.8

    Basic inspection lacks statistical power and is considered a less-than-ideal method.

  - Correlated Inspection: Compare performance measure under the assumption that the simulation experiences exactly the same random variables as the real system.

    Somewhat difficult to do in practice.

Basic Inspection

- Consider two M/M/1 systems one with ρ = .5 the other ρ = .6 (these are significantly different systems).
- Below are the results of 10 runs (of 200 jobs) in which we look at time in system.

Run	1	2	3	4	5	6	7	8	9	10
Sys 1 ρ = .5	.548	.491	.490	.454	.567	.486	.419	.527	.521	.461
Sys 2 ρ = .6	.613	.618	.630	.732	.548	.614	.463	.614	.463	.572
Diff	.065	.127	.140	.278	-.019	.128	.044	.087	-.058	.111

Basic Inspection

Run	1	2	3	4	5	6	7	8	9	10
Sys 1 $\rho = .5$	.548	.491	.490	.454	.567	.486	.419	.527	.521	.461
Sys 2 $\rho = .6$	.613	.618	.630	.732	.548	.614	.463	.614	.463	.572
Diff	.065	.127	.140	.278	-.019	.128	.044	.087	-.058	.111

Since S2 – S1 brackets 0, we might incorrectly conclude that these two systems are not different.

If we only looked at a single run, there is a 20% chance that we might incorrectly identify S2 as having a smaller time in system!

Correlated Inspection

- Law and Kelton suggest that we attempt to ensure that the simulation sees exactly the same jobs as the real system and then comparing the output.
- *(In this example, I used common random numbers on two simulations)*

Run	1	2	3	4	5	6	7	8	9	10
Sys 1 $\rho = .5$	.393	.528	.465	.583	.528	.574	.607	.503	.450	.486
Sys 2 $\rho = .6$	.472	.634	.558	.700	.633	.689	.728	.603	.540	.583
Diff	.079	.006	.093	.117	.105	.115	.121	.100	.090	.097

In practice this isn't always an easy technique to use. We may lack the data or the means to force historical data through the simulation.

Confidence Interval Approach

Once again, consider the two M/M/1 systems

Run	1	2	3	4	5	6	7	8	9	10
Sys 1 $\rho = .5$	.548	.491	.490	.454	.567	.486	.419	.527	.521	.461
Sys 2 $\rho = .6$	.613	.618	.630	.732	.548	.614	.463	.614	.463	.572
Diff	.065	.127	.140	.278	-.019	.128	.044	.087	-.058	.111

We could simply develop a confidence interval on S2-S1. If the confidence interval contains 0, we cannot say that the two results are different.

Confidence Interval Approach

Run	1	2	3	4	5	6	7	8	9	10
Sys 1 ρ = .5	.548	.491	.490	.454	.567	.486	.419	.527	.521	.461
Sys 2 ρ = .6	.613	.618	.630	.732	.548	.614	.463	.614	.463	.572
Diff	.065	.127	.140	.278	-.019	.128	.044	.087	-.058	.111

Mean (S2-S1): .0903
Var(S2-S1): .0086
s: .0930
$t_{9,.95}$: 1.833

$$CI: \overline{(S2-S1)} \pm t_{n-1,1-\alpha/2}\sqrt{\frac{Var(S2-S1)}{n}}$$

$$.0902 \pm 1.833(.0294)$$

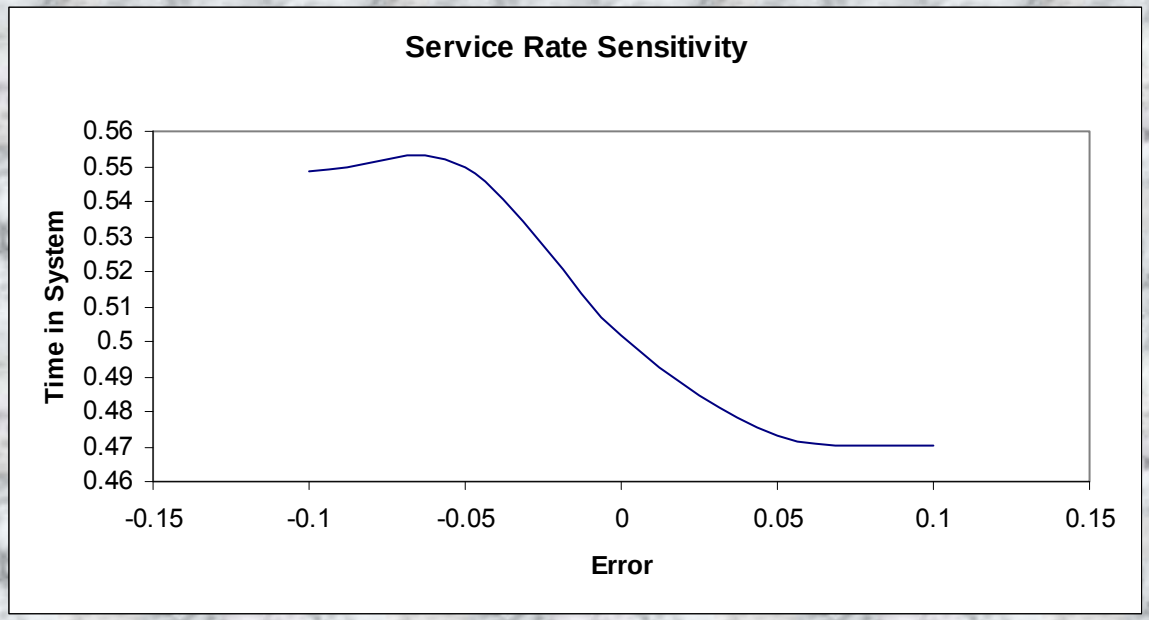
$$(.0365, 1.44)$$

Based on this test, we would assume that S2 and S1 are different.

A Plan for Validation - Operational

- **Robustness:** In our example the potential improvement of S1 over S2 is about 15%. Depending on our confidence in our data, this may (or may not) be a robust result.
- **Sensitivity:**

In this example, the model is insensitive to positive errors (faster service) but sensitive to negative errors (slower service)



A Plan for Validation - Dynamic

- **Maintainability**: This is harder to prove, but is a function of good documentation, good software, and a good interface design.
- **Review and Update Processes**: Require written documentation defining a plan for how model is to be reviewed and when it will be updated.

Documentation

User manual

Code documentation

Interface design document.

Review Plan

This model will be reviewed quarterly by IE.

The values of λ&μ will be recalculated…

Update Plan

Updates will be conducted quarterly or as needed.

IE will be responsible.

Credibility

- The ultimate goal is to build models that are credible with decision makers.
- Keep decision makers informed.
- Interact with them frequently.
- Keep good records and documentation. Detail your validation efforts.

A Final Word

- Almost all validation approaches assume the existence of a “real world” system to benchmark your model.
- When no such system exists, you must be very methodical in your attempts to validate.
- The Schellenberger framework can still be used and should guide your efforts.

References

1. *Modeling and Simulation Verification and Validation Challenges* - Dale K. Pace.
2. *How to Perform Credible Verification,Validation, and Accreditation for Modeling and Simulation* - Dr. David A. Cook and Dr. James M. Skinner, The AEgis Technologies Group, Inc.
3. *Model Verification and Validation* - Charles M. Macal*
4. Verification and Validation of Simulation Models - *Verification and Validation of Simulation Models*
5. *DoD Modeling and Simulation (M&S) Verification, Validation, and Accreditation (VV&A)* - Department of Defense
6. *Verification & Validation - Credibility in Stockpile Modeling and Simulation* - http://www.sandia.gov/NNSA/ASC/factSHT6.pdf.
7. *Modeling and Simulation Verification, Validation, and Accreditation - Implementation Handbook* - Navy Modeling and Simulation Management Office