UNIT 8: Verification and validation modeling

- One of the most important and difficult tasks facing a model developer is the Verification and validation of the simulation model.
- It is the job of the model developer to work closely with the end users Throughout the period (development and validation to reduce this skepticism And to increase the credibility.

The goal of the validation process is twofold:

1: To produce a model that represents true system behavior closely enough for the model to be used as a substitute for the actual system for the purpose of experimenting with system.

2: To increase an acceptable, level the credibility of the model ,so that the model will be used by managers and other decision makers.|

The verification and validation process consists of the following components:-

1:Verification is concerned with building the model right. It is utilized in comparison of the conceptual model to the computer representation that implements that conception. It asks the questions: Is the model implemented correctly in the computer? Are the input parameters and logical structure of the model correctly represented?

2: Validation is concerned with building the right model. It is utilized to determine that a model is an accurate representation of the real system. It is usually achieved through the calibration of the model

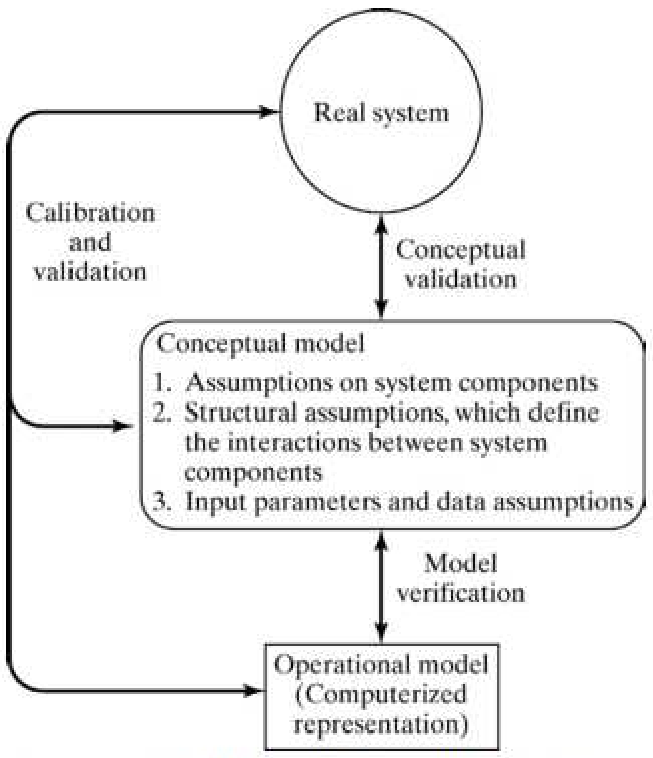

Real system
Calibration and validation
Conceptual validation
Conceptual model
1. Assumptions on system components
2. Structural assumptions, which define the interactions between system components
3. Input parameters and data assumptions
Model verification
Operational model (Computerized representation)

Figure 1 Model building, verification, and validation

7.2 Verification of Simulation Models

- The purpose of model verification is to assure that the conceptual model is reflected accurately in the computerized representation.
- The conceptual model quite often involves some degree of abstraction about system operations, or some amount of simplification of actual operations.

Many common-sense suggestions can be given for use in the verification process:-

- Have the computerized representation checked by someone other than its developer.
- Make a flow diagram which includes each logically possible action a system can take when an event occurs, and follow the model logic for each a for each action for each event type.
- Closely examine the model output for reasonableness under a variety of settings of Input parameters.
- Have the computerized representation print the input parameters at the end of the Simulation to be sure that these parameter values have not been changed inadvertently.
- Make the computerized representation of self-documenting as possible.
- If the computerized representation is animated, verify that what is seen in the animation imitates the actual system.
- The interactive run controller (IRC) or debugger is an essential component of Successful simulation model building. Even the best of simulation analysts makes mistakes or commits logical errors when building a model.

  The IRC assists in finding and correcting those errors in the follow ways:

  (a) The simulation can be monitored as it progresses.

  (b) Attention can be focused on a particular line of logic or multiple lines of logic that constitute a procedure or a particular entity.

  (c) Values of selected model components can be observed. When the simulation has paused, the current value or status of variables, attributes, queues, resources, counters, etc., can be observed

  (d) The simulation can be temporarily suspended, or paused, not only to view information but also to reassign values or redirect entities.

- Graphical interfaces are recommended for accomplishing verification & validation

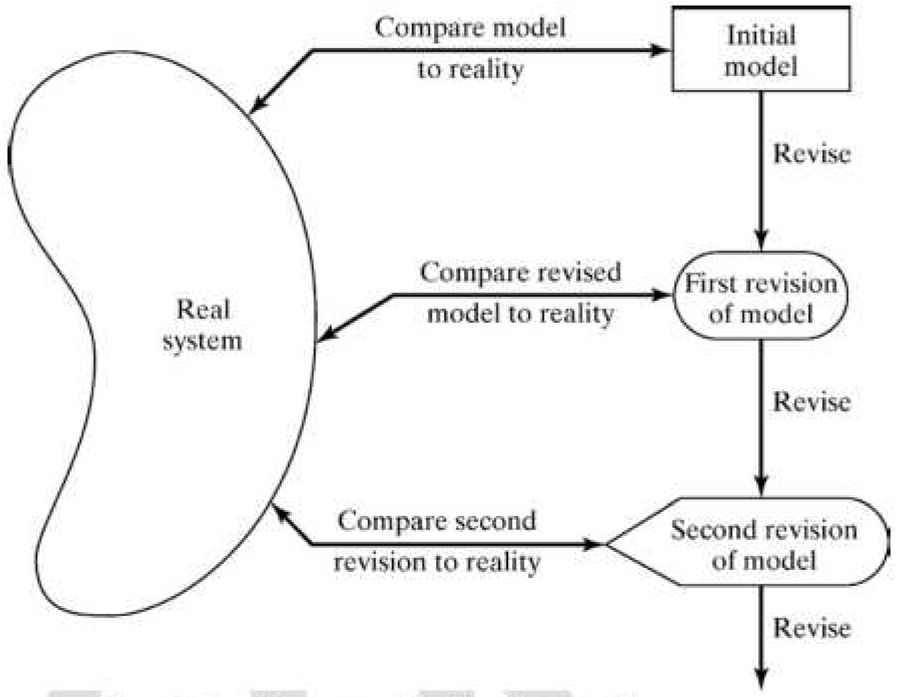


Figure 2 Iterative process of calibration a model

As an aid in the validation process, Naylor finger:

1. Build a model that has high face validity.
2. Validate model assumption.
3. Compare the model input-output transformation to cooresponding input-output transformation for the real system.

7.3.1 FACE VALIDITY

- The first goal of the simulation modeler is to construct a model that appears reasonable on its face to model users and others who are knowledgeable about the real system being simulated.
- The users of a model should be involved in model construction from its conceptualization to its implementation to ensure that a high degree of realism is built into the model through reasonable assumptions regarding system structure, and reliable data.
- Another advantage of user involvement is the increase in the models perceived validity or credibility without which manager will not be willing to trust simulation results as the basis for decision making.
- Sensitivity analysis can also be used to check model's face validity.
- The model user is asked if the model behaves in the expected way when one or more input variables is changed.
- Based on experience and observations on the real system the model user and model builder would probably have some notion at least of the direction of change in model output when an input variable is increased or decreased.
- The model builder must attempt to choose the most critical input variables for testing if it is too expensive or time consuming to: vary all input variables

7.3.2 Validation of Model Assumptions

- Model assumptions fall into two general classes: structural assumptions and data assumptions.
- Structural assumptions involve questions of how the system operates and usually involve simplification and abstractions of reality.

- For example, consider the customer queuing and service facility in a bank. Customers may form one line, or there may be an individual line for each teller. If there are many lines, customers may be served strictly on a first-come, first-served basis, or some customers may change lines if one is moving faster.
- The number of tellers may be fixed or variable. These structural assumptions should be verified by actual observation during appropriate time periods together with discussions with managers and tellers regarding bank policies and actual implementation of these policies.
- Data assumptions should be based on the collection of reliable data and correct statistical analysis of the data.data were collected on:

  1. Inter arrival times of customers during several 2-hour periods of peak loading ("rush-hour" traffic)
  2. Inter arrival times during a slack period
  3. Service times for commercial accounts
  4. Service times for personal accounts

- Validation is not an either/or proposition—no model is ever totally representative of the system under study. In addition, each revision of the model, as in the Figure above involves some cost, time, and effort.
- The procedure for analyzing input data consist of three steps:-

  1: Identifying the appropriate probability distribution.

  2: Estimating the parameters of the hypothesized distribution .

  3: Validating the assumed statistical model by goodness – of – fit test such as the chi square test, KS test and by graphical methods

10.3.3 Validating Input-Output Transformation

- In this phase of validation process the model is viewed as input –output transformation.
- That is, the model accepts the values of input parameters and transforms these inputs into output measure of performance. It is this correspondence that is being validated.
- Instead of validating the model input-output transformation by predicting the future ,the modeler may use past historical data which has been served for validation purposes that

is, if one set has been used to develop calibrate the model, its recommended that a separate data test be used as final validation test.

- Thus accurate " prediction of the past" may replace prediction of the future for purpose of validating the future.
- A necessary condition for input-output transformation is that some version of the system under study exists so that the system data under at least one set of input condition can be collected to compare to model prediction.
- If the system is in planning stage and no system operating data can be collected, complete input-output validation is not possible.
- Validation increases modeler's confidence that the model of existing system is accurate.
- Changes in the computerized representation of the system, ranging from relatively minor to relatively major include :

1: Minor changes of single numerical parameters such as speed of the machine, arrival rate of the customer etc.

2: Minor changes of the form of a statistical distribution such as distribution of service time or a time to failure of a machine.

3: Major changes in the logical structure of a subsystem such as change in queue discipline for waiting-line model, or a change in the scheduling rule for a job shop model.

4: Major changes involving a different design for the new system such as computerized inventory control system replacing a non computerized system .

- If the change to the computerized representation of the system is minor such as in items one or two these change can be carefully verified and output from new model can be accepted with considerable confidence.

7.3.4: Input-Output Validation: Using Historical Input Data

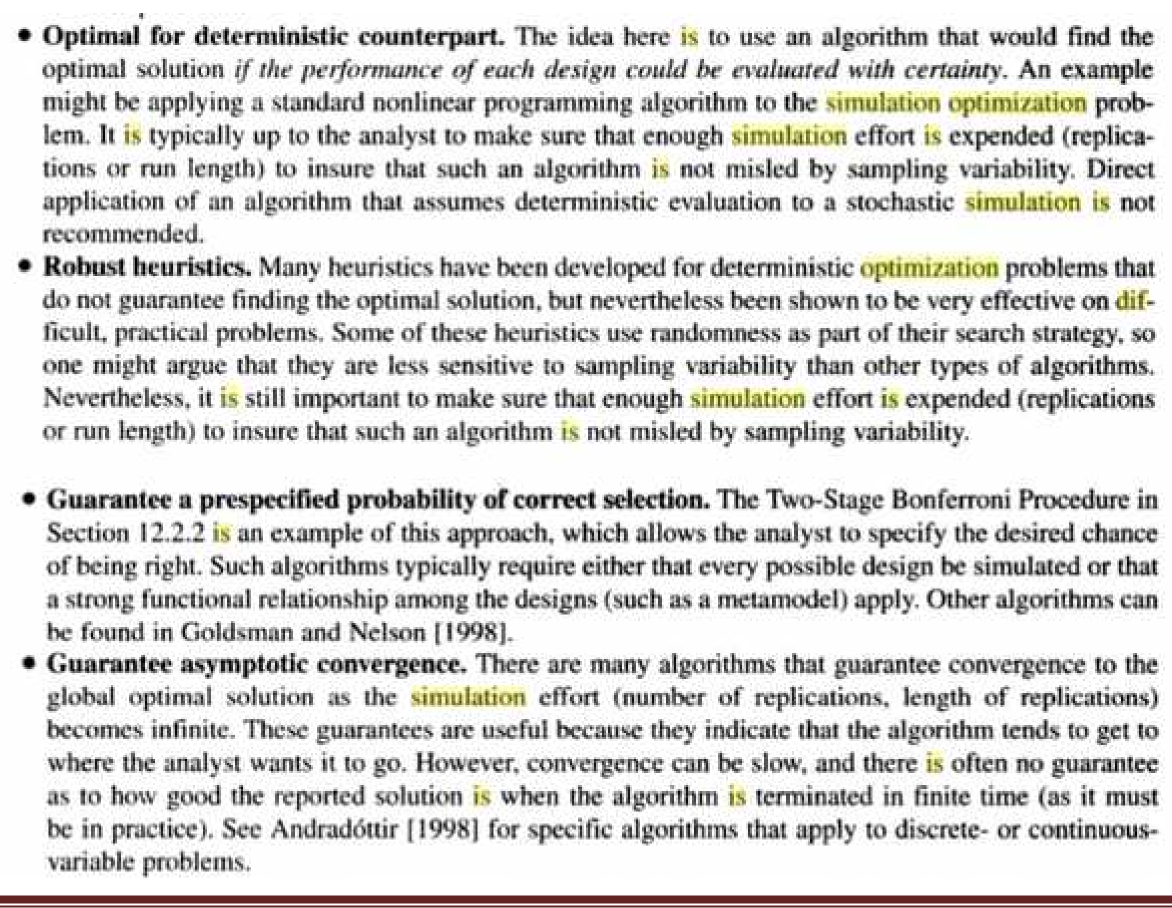
- When using artificially generated data as input data the modeler expects the model produce event patterns that are compatible with, but not identical to, the event patterns that occurred in the real system during the period of data collection.
- Thus, in the bank model, artificial input data {X\n, X2n, n = 1,2, , .} for inter arrival and service

times were generated and replicates of the output data Y2 were compared to what was observed in the real system

- An alternative to generating input data is to use the actual historical record, {An, Sn, n = 1,2,...}, to drive simulation model and then to compare model output to system data.
- To implement this technique for the bank model, the data Ai, A2,..., S1 S2 would have to be entered into the model into arrays, or stored on a file to be read as the need arose.
- To conduct a validation test using historical input data, it is important that all input data (An, Sn,...) and all the system response data, such as average delay(Z2), be collected during the same time period.
- Otherwise, comparison of model responses to system responses, such as the comparison of average delay in the model (Y2) to that in the system (Z2), could be misleading.
- responses (Y2 and 22) depend on the inputs (An and Sn) as well as on the structure of the system, or model.
- Implementation of this technique could be difficult for a large system because of the need for simultaneous data collection of all input variables and those response variables of primary interest.

7.3.5: Input-Output Validation: Using a Turing Test

- In addition to statistical tests, or when no statistical test is readily applicable persons knowledgeable about system behavior can be used to compare model output to system output.
- For example, suppose that five reports of system performance over five different days are prepared, and simulation output are used to produce five "fake" reports. The 10 reports should all be in exactly in the same format and should contain information of the type that manager and engineer have previously seen on the system.
- The ten reports are randomly shuffled and given to the engineers, who is asked to decide which report are fake and which are real.
- If engineer identifies substantial number of fake reports the model builder questions the engineer and uses the information gained to improve the model.
- If the engineer cannot distinguish between fake and real reports with any consistency, the modeler will conclude that this test provides no evidence of model inadequacy .



- **Optimal for deterministic counterpart.** The idea here is to use an algorithm that would find the optimal solution *if the performance of each design could be evaluated with certainty*. An example might be applying a standard nonlinear programming algorithm to the simulation optimization problem. It is typically up to the analyst to make sure that enough simulation effort is expended (replications or run length) to insure that such an algorithm is not misled by sampling variability. Direct application of an algorithm that assumes deterministic evaluation to a stochastic simulation is not recommended.
- **Robust heuristics.** Many heuristics have been developed for deterministic optimization problems that do not guarantee finding the optimal solution, but nevertheless been shown to be very effective on difficult, practical problems. Some of these heuristics use randomness as part of their search strategy, so one might argue that they are less sensitive to sampling variability than other types of algorithms. Nevertheless, it is still important to make sure that enough simulation effort is expended (replications or run length) to insure that such an algorithm is not misled by sampling variability.

- **Guarantee a prespecified probability of correct selection.** The Two-Stage Bonferroni Procedure in Section 12.2.2 is an example of this approach, which allows the analyst to specify the desired chance of being right. Such algorithms typically require either that every possible design be simulated or that a strong functional relationship among the designs (such as a metamodel) apply. Other algorithms can be found in Goldsman and Nelson [1998].
- **Guarantee asymptotic convergence.** There are many algorithms that guarantee convergence to the global optimal solution as the simulation effort (number of replications, length of replications) becomes infinite. These guarantees are useful because they indicate that the algorithm tends to get to where the analyst wants it to go. However, convergence can be slow, and there is often no guarantee as to how good the reported solution is when the algorithm is terminated in finite time (as it must be in practice). See Andradóttir [1998] for specific algorithms that apply to discrete- or continuous-variable problems.

Output Analysis for Steady-State Simulations I

- Consider a single run of a simulation model to estimate a steadystate or long-run characteristics of the system.
- The single run produces observations $Y_1, Y_2, \ldots$ (generally the samples of an autocorrelated time series).
- Performance measure:

$$\theta = \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} Y_i, \qquad \text{for discrete measure}$$

$$\phi = \lim_{T_E\to\infty} \frac{1}{T_E}\int_0^{T_E} Y(t)dt, \qquad \text{for continuous measure}$$

independent of initial conditions, both with probability 1

- The sample size is a design choice, with several considerations in mind:

Initialization Bias I

- Methods to reduce the point-estimator bias caused by using artificial and unrealistic initial conditions:
  - Intelligent initialization.
  - Divide simulation into an initialization phase and data collection phase.
- Intelligent initialization
  - Initialize the simulation in a state that is more representative of long-run conditions.
  - If the system exists, collect data on it and use these data to specify more nearly typical initial conditions.
  - If the system can be simplified enough to make it mathematically solvable, e.g. queueing models, solve the simplified model to find long-run expected or most likely conditions, use that to initialize the simulation.
- Divide each simulation into two phases:
  - An initialization phase, from time 0 to time T_0.

Error Estimation I

- If $\{Y_1, \ldots, Y_n\}$ are not statistically independent, then S^2/n is a biased estimator of the true variance.
- Almost always the case when $\{Y_1, \ldots, Y_n\}$ is a sequence of output observations from within a single replication (autocorrelated sequence, time-series).
- Suppose the point estimator $\widehat{\theta}$ is the sample mean

$$\overline{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i$$

- Variance of $\overline{Y}$ is very hard to estimate.
- For systems with steady state, produce an output process that is approximately covariance stationary (after passing the transient phase).

Replication Method I

- Use to estimate point-estimator variability and to construct a confidence interval.
- Approach: make R replications, initializing and deleting from each one the same way.
- Important to do a thorough job of investigating the initial-condition bias:
  - Bias is not affected by the number of replications, instead, it is affected only by deleting more data (i.e., increasing T_0) or extending the length of each run (i.e. increasing T_E).
- Basic raw output data $\{Y_{rj}, r = 1, \dots, R, \ j = 1, \dots, n\}$ is derived by:
  - Individual observation from within replication r.
  - Batch mean from within replication r of some number of discrete-time observations.

Sample Size I

- To estimate a long-run performance measure, θ, within $\pm\varepsilon$ with confidence $100(1-\alpha)\%$.
- M/G/1 queueing example (cont.):
  - We know: $R_0 = 10$, $d = 2$ deleted and $S_0^2 = 25.30$.
  - To estimate the long-run mean queue length, L_Q, within $\varepsilon = 2$ customers with 90% confidence ($\alpha = 10\%$).
  - Initial estimate:

$$R \geq \left(\frac{z_{0.05}S_0}{\varepsilon}\right)^2 = \frac{1.645^2(25.30)}{2^2} = 17.1$$

  - Hence, at least 18 replications are needed, next try $R = 18, 19, \dots$ using $R \geq (t_{0.05,R-1}S_0/\varepsilon)^2$. We found that

$$R = 19 \geq (t_{0.05,R-1}S_0/\varepsilon)^2 = \left(1.73^2 \cdot 25.3/4\right) = 18.93$$

  - Additional replications needed is $R - R_0 = 19 - 10 = 9$.

Batch Means for Interval Estimation

- Using a single, long replication:
  - Problem: data are dependent so the usual estimator is biased.
  - Solution: batch means.
- Batch means: divide the output data from 1 replication (after appropriate deletion) into a few large batches and then treat the means of these batches as if they were independent.
- A continuous-time process, $\{Y(t), T_0 \le t \le T_0 + T_E\}$:
  - k batches of size $m = T_E/k$, batch means:

$$\overline{Y}_j = \frac{1}{m}\int_{(j-1)m}^{jm} Y(t+T_0)dt, \; j = 1, 2, \ldots, k$$

  - A discrete-time process, $\{Y_i, i = d+1, d+2, \ldots, n\}$:
    - k batches of size $m = (n-d)/k$, batch means:

$$\overline{Y}_j = \frac{1}{m}\sum_{i=(j-1)m+1}^{jm} Y_{i+d}, \qquad j = 1, 2, \ldots, k$$