ZENITH STUDY HUB

Modeling and Simulation

UNIT III | INPUT MODELING

Measured data, statistical distributions, goodness of fit, pseudorandom numbers and simulation inputs

SPPU TE Robotics & Automation | 2024 Pattern | PEC321-ROA (B)

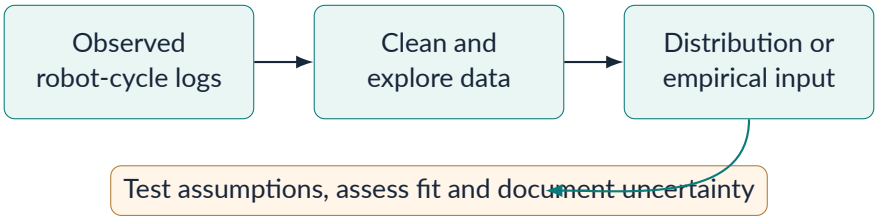

Figure 1. Original evidence-to-model workflow. A fitted input model is a testable representation of data, not automatically the physical truth.

Official scope
SPPU TE Robotics & Automation 2024 Pattern, PEC321-ROA (B), Unit III, 8 hours. Topics verified against the supplied official *TE Syllabus.pdf*, physical PDF p. 27. Illustrative datasets in this PDF are synthetic and are not taken from real factory measurement logs.

Study edition: 21 November 2026 | Author: Om Deshmukh | Detailed review candidate

Complete Unit III syllabus coverage

Official topic	How this guide develops it
Nature of input data	Observed vs assumed inputs, units, time stamps, dependencies
Data collection methods/sources	PLC, sensors, MES, time studies, selection bias
Preprocessing	Missingness, duplicates, outliers, clock problems, unit consistency
Identifying distributions	Physical mechanisms, discrete/continuous families, empirical models
Histogram analysis	Real bins from a 12-observation teaching sample
Goodness-of-fit tests	Chi-square statistic, expected counts, KS and test limits
Testing assumptions	Stationarity, independence, autocorrelation, subgroup effects
Introduction to random numbers	Uniform samples, simulation seed and repeatability
Properties of random numbers	Uniformity, independence, period, reproducibility
Pseudorandom number generators	Deterministic state-update methods and stream control
Congruential generators	Complete 16-state toy LCG and limitations
Randomness testing	Frequency, correlation, run tests and statistical caveats
Applications in simulation	Robot cycle times, AGV dispatch, downtime and queue arrivals

In simple language

Input modeling means turning observations about the real world into a carefully justified input process for a simulation. **Random-number generation** means producing reproducible computer numbers that let a stochastic model explore different possible outcomes. The two tasks are related but not identical: good random numbers cannot repair a bad fitted model.

Check your understanding

Do not copy a probability distribution merely because it is common in textbooks. First ask whether the data source, observation period, sampling design and physical mechanism justify that assumption.

1 Nature of Simulation Input Data

A simulation contains inputs describing its operating conditions. Examples include part interarrival times, robot handling durations, tool-breakdown events, operator attendance, equipment repair times, batch sizes and AGV trip distances. Some parameters are measured directly and others are assumed. The model should distinguish them.

1.1 Types of input data

Discrete input: a count or category, such as three incoming parts per pallet or a binary machine-failure indicator. **Continuous input:** a measurement, such as handling time or travel distance. **Time-series input:** repeated observations indexed by time, where adjacent values may be related. **Event log:** a record of changes such as job release, service start, breakdown and return to service.

Quantity	Correct representation and caution
Interarrival time (s)	Positive durations between ordered arrival timestamps
Robot cycle time (s)	Service times; may depend on part type, gripper or shift
Breakdown count	Discrete events in an explicitly stated observation window
Repair time (min)	Positive durations, possibly censored by end of shift
Part category	Categorical mix, potentially related to cycle time
Sensor current (A)	Measured numerical signal with calibration and resolution

In simple language
A *random variable* is not the same as a random-number generator. The variable describes an uncertain quantity in the engineering model; a generator creates numerical samples used to imitate its probability law.

1.2 Measured data versus assumptions

Suppose a robot handles one part every 8 seconds in a vendor demonstration. Entering a fixed value of 8 s is an assumption about the production process, not

evidence that all future jobs last exactly 8 s. A real shift may include changing products, micro-stops and gripper retries. Input uncertainty and measurement error should both be documented.

2 Collecting Reliable Engineering Data

2.1 Common sources and collection methods

- **PLC/SCADA logs:** machine state, event timestamps, faults and cycle counters. Confirm clock accuracy and whether events are lost during network outages.
- **MES or production records:** part types, work orders, output and downtime codes. Reconcile identifiers with station-level logs.
- **Time study and video observation:** manually observe a stratified sample of job types, not only the easiest shift.
- **Instrumented robot trials:** obtain motion times, stops and retry counts, while identifying warm-up and test-condition effects.
- **Supplier specifications:** useful for limits and first assumptions, but not automatically representative of a particular factory.

2.2 A practical data dictionary

Field	Units/type	Validation question
`job_id`	identifier	Unique per attempted part?
`start_ts`	timestamp	Same clock as completion?
`end_ts`	timestamp	End after start, including crossing midnight?
`product_type`	category	Consistent codebook?
`tool_retry`	nonnegative count	Recorded even when the job fails?
`stop_reason`	category	Is a missing reason a true “none”?

Engineering collection rule
Preserve raw data and metadata. Keep a reversible cleaning script and a record of excluded observations. That way another engineer can reproduce exactly how the model inputs were obtained.

2.3 Representativeness and observation windows

A dataset gathered only during a smooth morning shift may be biased toward high productivity. A data sample gathered after a major maintenance event may miss older-equipment failure behavior. Separate operating conditions before pooling observations. Censored events, such as a repair still incomplete at the end of a shift, require special treatment rather than assigning them an invented duration.

3 Data Preprocessing Before Fitting

Data cleaning should improve measurement integrity without erasing genuine production behavior.

1. **Check units.** Convert milliseconds and seconds deliberately and record the original unit.
2. **Sort timestamps.** Handle clock synchronization, daylight-saving conventions, duplicate messages and event ordering.
3. **Identify impossible records.** A negative duration often indicates a data error; investigate rather than accepting it blindly.
4. **Distinguish missing from zero.** A missing cycle time is not a completed zero-second cycle.
5. **Inspect outliers in context.** A very long cycle can be a real jam, repair or rare event and is often important to simulation.
6. **Segment relevant subgroups.** Product, robot, operator, shift and maintenance state may explain important differences.
7. **Document all changes.** Record row counts and reasons for every exclusion, interpolation or correction.

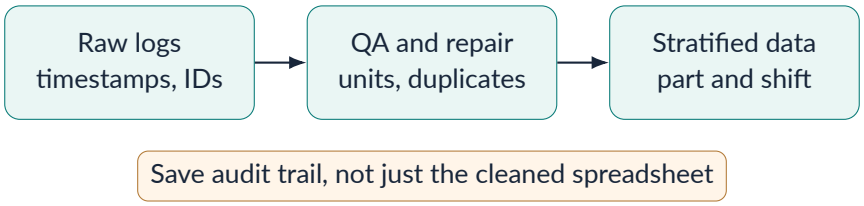

Figure 2. Original data-preprocessing flow. Grouping by operating condition helps avoid mixing fundamentally different process distributions.

Check your understanding
Never discard every observation above a fixed threshold simply to obtain a tidy bell curve. Investigate whether the tail represents actual jams, delays or fault recovery, which may dominate queue lengths and due-date risks.

4 Exploratory Data Analysis and Histograms

A histogram groups observed values into intervals and displays their counts or densities. The appearance depends on sample size, units and bin width. A histogram is an exploratory visual, not a statistical proof of distribution shape.

4.1 Worked example: 12 measured-like service times

Use the *illustrative* seconds dataset

$$(2, 3, 2, 4, 3, 3, 5, 2, 4, 4, 3, 5).$$

There are 12 observations, with counts for durations 2, 3, 4 and 5 seconds equal to 3, 4, 3 and 2 respectively.

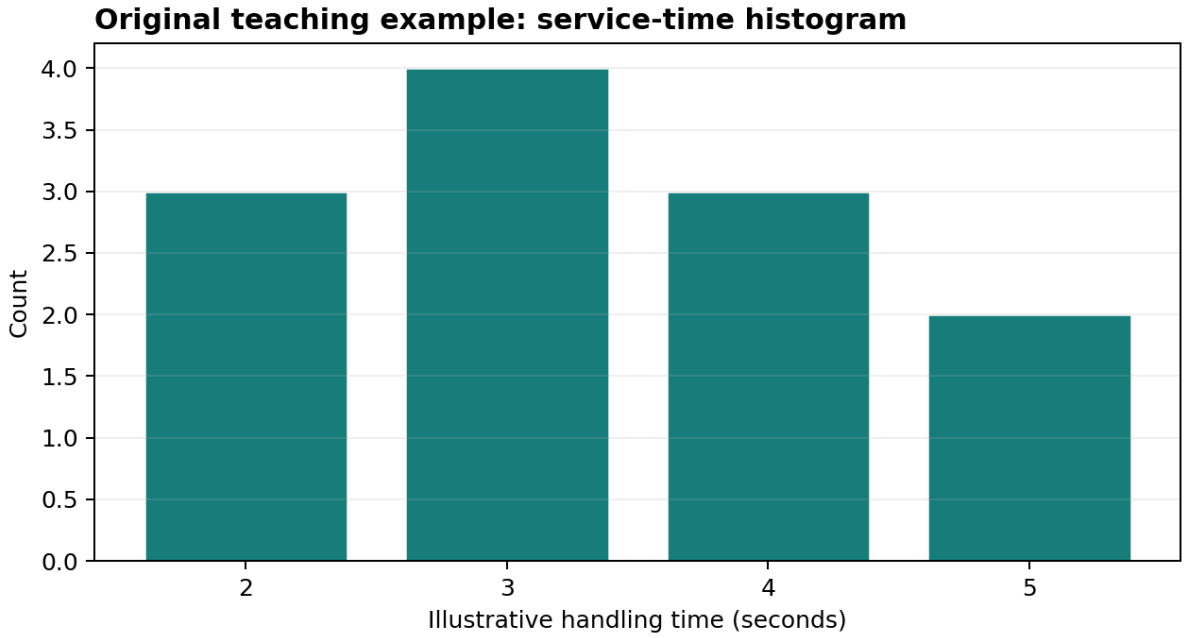

Figure 3. Original synthetic-data histogram. Frequencies are 3, 4, 3, 2; 12 observations in total. A smooth fitted curve would not be justified solely from this small sample.

4.2 Recompute its statistics

The sum is 40 seconds, so the mean is $\bar{x} = 40/12 = 3.333\ldots$ s. The sorted values are $2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5$, so the median is 3 s and the mode is 3 s. Both the mean and the shape depend on the mix of short and long jobs; that is why reporting one average is insufficient.

Sample variance, step by step

Use the frequency table rather than writing all twelve squares separately. The sum of squares is $3(2^2) + 4(3^2) + 3(4^2) + 2(5^2) = 146$. The corrected sample variance is

$$s^2 = \frac{\sum x_i^2 - n\bar{x}^2}{n-1} = \frac{146 - 12(10/3)^2}{11} = \frac{38}{33} \approx 1.1515 \text{ s}^2.$$

Thus the sample standard deviation is $s \approx 1.0731$ s. These statistics describe the example sample, not an established physical probability law.

4.3 An empirical cumulative distribution

An empirical cumulative distribution function (ECDF) is the fraction of observations at or below a threshold:

$$\widehat{F}_n(t) = \frac{\#\{x_i \leq t\}}{n}.$$

For the 12 observations, seven are at most 3 seconds, so $\widehat{F}_n(3) = 7/12 \approx 0.5833$. An ECDF preserves visible step structure and is helpful when a textbook distribution looks unjustified.

5 Identifying Candidate Probability Distributions

First reason about the physics and data type, then estimate candidate parameters and compare the fitted model with the observations. A positive cycle time cannot be described by an unconstrained negative-valued distribution without explaining truncation or another transformation.

Possible model	Engineering interpretation and warnings
Empirical discrete	Use sampled classes or bins directly; preserves supported data but may miss unobserved tails
Uniform	Every value in a declared finite range equally likely; rarely a default for real service times
Exponential	Memoryless positive waiting times under a suitable process mechanism; not automatically correct for robot service times
Poisson	Event counts over a fixed interval under a suitable constant-rate, independent-event assumption
Normal	Approximately symmetric, unbounded distribution; can assign physically impossible negative times
Lognormal	Positive, right-skewed duration when multiplicative mechanisms are plausible
Weibull	Flexible positive failure-time or lifetime model; needs parameter fitting and diagnostic checks

5.1 Event counts versus interarrival times

A Poisson model describes counts in an interval; its associated homogeneous Poisson process has exponentially distributed waiting times. Do not substitute one for the other or assume arrivals are Poisson merely because the process is random. Scheduled job releases, shift changes and batch arrivals can violate the model.

Dimensional reasoning: a hypothesized arrival rate
If the hypothesized constant arrival rate is 12 jobs/hour, the mean exponential waiting time would be $1/12$ hour = 5 minutes. Under that assumption, $P(T > 10 \text{ min}) = e^{-10/5} = e^{-2} \approx 0.1353$. This is a consequence of the chosen model, not a claim about measured arrival patterns.

5.2 Inverse-transform sampling

If U is uniform on $(0, 1)$ and the hypothesized waiting-time CDF is $F(t) = 1 - e^{-\lambda t}$, setting $F(T) = U$ gives

$$T = -\frac{\ln(1 - U)}{\lambda}.$$

For $\lambda = 0.2$ per minute and $U = 0.5$, $T = \ln 2/0.2 \approx 3.466$ min. Avoid evaluating $\ln(0)$ when $U = 1$; implementation details must respect the generator's endpoint conventions.

6 Goodness-of-Fit Tests: Meaning and Mechanics

A goodness-of-fit (GOF) test asks whether the observed discrepancy between a candidate distribution and data is unusually large under a stated null hypothesis. It does *not* prove that the hypothesis is physically correct. Small samples may have low power; huge samples may reject small, unimportant deviations.

6.1 Chi-square goodness-of-fit

With observed counts O_j and model-based expected counts $E_j > 0$ for k categories,

$$\chi^2 = \sum_{j=1}^{k} \frac{(O_j - E_j)^2}{E_j}.$$

The chi-square approximation needs adequate expected counts. The correct degrees of freedom depend on the number of bins and on whether parameters were estimated from the same data. For pre-specified probabilities, degrees of freedom are commonly $k - 1$; estimating p fitted parameters under regular conditions may reduce this to $k - 1 - p$.

Worked 40-observation four-bin example

Suppose a *fully specified* uniform four-category model predicts ten counts per bin in 40 independent trials. Actual counts are $9, 11, 10, 10$. Then

$$\chi^2 = \frac{(9-10)^2}{10} + \frac{(11-10)^2}{10} + 0 + 0 = 0.2.$$

With four pre-specified bins and no parameters fitted from these observations, the reference degrees of freedom would be 3. The small statistic is compatible with this hypothesized model; it does not prove that the random process is uniform or independent.

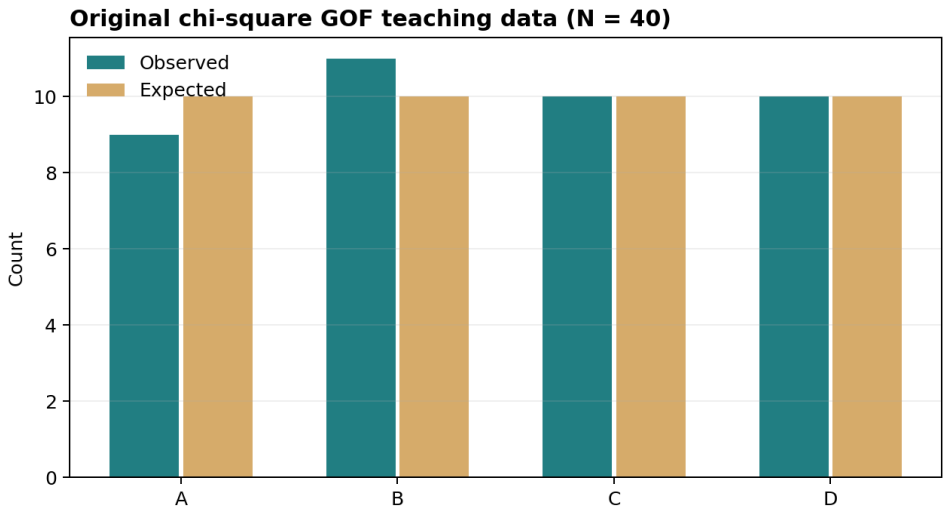

Figure 4. Original chi-square counts. The figure shows $O = (9, 11, 10, 10)$ and $E = (10, 10, 10, 10)$; $\chi^2 = 0.2$ for this synthetic experiment.

6.2 Why the chi-square result can mislead

Changing how observations are binned can change the statistic and power. Small expected counts weaken the usual approximation. Estimating a distribution's parameters from the same sample affects degrees of freedom and, for some approaches, the null distribution. The assumptions and binning scheme must be disclosed.

Check your understanding
Interpretation trap: "We did not reject the candidate distribution" is not the same as "We proved this distribution." Also check independence and whether the sample represents the operating conditions you intend to simulate.

6.3 Kolmogorov–Smirnov (KS) testing

For a fully specified continuous CDF F, the classical one-sample KS statistic is

$$D_n = \sup_x \left| \widehat{F}_n(x) - F(x) \right| .$$

It looks at the maximum vertical gap between the empirical and candidate cumulative distributions. The classical null reference distribution is not generally valid when parameters were fitted to the same sample; corrections or calibration may be required. The textbook KS test also assumes independent observations and a continuous, fully specified null distribution. NIST highlights these limitations.

Which test should I use?
Use a visual inspection, the model's physical justification and appropriate diagnostics together. A chi-square test suits suitably populated binned counts; a KS-style test evaluates CDF discrepancies for continuous distributions under its relevant conditions. Neither replaces checking dependence, shifts, missing records or sensor calibration.

7 Testing Input Assumptions

Distribution shape alone is insufficient. A sequence with correct marginal frequencies but strong time dependence can create very different queue and reli-

ability outputs.

7.1 Independence versus stationarity

Independence means observing one value provides no statistical information about another under the model. **Stationarity**, in the commonly intended weaker sense, means key distributional characteristics are stable over the chosen time horizon. They are distinct: a sequence can have the same histogram over time while adjacent values remain correlated.

7.2 Worked lag-one autocorrelation

Use the illustrative sequence $x = (2, 2, 3, 3, 4, 4)$ with mean $\bar{x} = 3$. Let $d_i = x_i - 3 = (-1, -1, 0, 0, 1, 1)$. A simple sample autocorrelation estimate at lag one is

$$r_1 = \frac{\sum_{i=1}^{5} d_i d_{i+1}}{\sum_{i=1}^{6} d_i^2} = \frac{2}{4} = 0.5.$$

This positive value illustrates neighbouring similar observations. It is a small synthetic sample, so no significance claim is justified. Note the precise autocorrelation estimator convention; other estimators normalize by different effective sample sizes.

7.3 Different products can have different processes

Suppose an illustrative robot station has product A cycle times $(3, 3, 4, 3)$ seconds and product B times $(9, 10, 11, 10)$ seconds. The subgroup means are 3.25 s and 10 s. The pooled mean is $53/8 = 6.625$ s, which describes neither product well. A simulation using only the pooled value can miss product-mix effects and queue bursts.

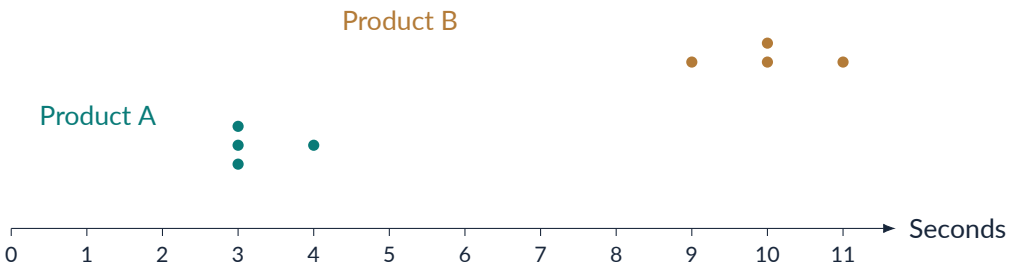

Figure 5. Original measured-like dot positions from the synthetic teaching dataset. Group A clusters around 3–4 s; group B around 9–11 s.

7.4 Practical assumption checks

Plot time against observation order; inspect missingness and shift boundaries; calculate autocorrelation cautiously; compare product groups; perform sensitivity experiments; and use a hold-out dataset when possible. If a new operating state starts after maintenance, fitting one stationary distribution across the change can hide the very effect the study should capture.

8 Introduction to Random-Number Generation

Stochastic simulations need a source of numbers that behaves suitably for the experiment. A **pseudorandom number generator** (PRNG) is a deterministic algorithm that evolves an internal state and produces a sequence that passes selected statistical properties. A seed initializes the state and enables a reproducible experimental run under a specified algorithm and software environment.

Property	Why it matters
Approximately uniform output	Mapping to probability intervals works as intended
Low unwanted dependence	Generated quantities should not create artificial patterns
Long period	The state sequence should not repeat too quickly
Reproducibility	Repeated experiments can be debugged and audited
Stream separation	Different replications avoid unintended overlapping sequences
Known algorithm and seed	A reviewer can reproduce the actual experiment

Security caution
Simulation PRNGs are not automatically cryptographically secure. Their suitability for Monte Carlo does not mean they can safely generate passwords, security tokens or encryption keys.

8.1 Uniform samples to model inputs

If the fitted input CDF is F, a uniform sample can often be transformed through an inverse or other sampling algorithm to produce X with the desired law. This mapping does not validate F. The input-modeling decision must be made on observational evidence independently from the PRNG selection.

9 Congruential Generators: Full Numerical Trace

A linear congruential generator (LCG) updates an integer state by

$$X_{n+1} = (aX_n + c) \bmod m, \quad U_n = X_n/m.$$

The multiplier a, increment c, modulus m and seed X_0 define the sequence. Here mod$m$ means take the remainder after division by m, so every $X_n \in \{0, \ldots, m-1\}$.

Worked LCG: $a = 5, c = 3, m = 16, X_0 = 1$
$X_1 = (5 \cdot 1 + 3) \bmod 16 = 8$. $X_2 = (5 \cdot 8 + 3) \bmod 16 = 11$. $X_3 = (5 \cdot 11 + 3) \bmod 16 = 10$.
Continue until the seed recurs. The states before the first repeat are

$$1, 8, 11, 10, 5, 12, 15, 14, 9, 0, 3, 2, 13, 4, 7, 6.$$

The next update from 6 is $(5 \cdot 6 + 3) \bmod 16 = 1$: the seed repeats after 16 steps, so this tiny generator's period is 16. *A maximal period does not demonstrate high randomness quality.*

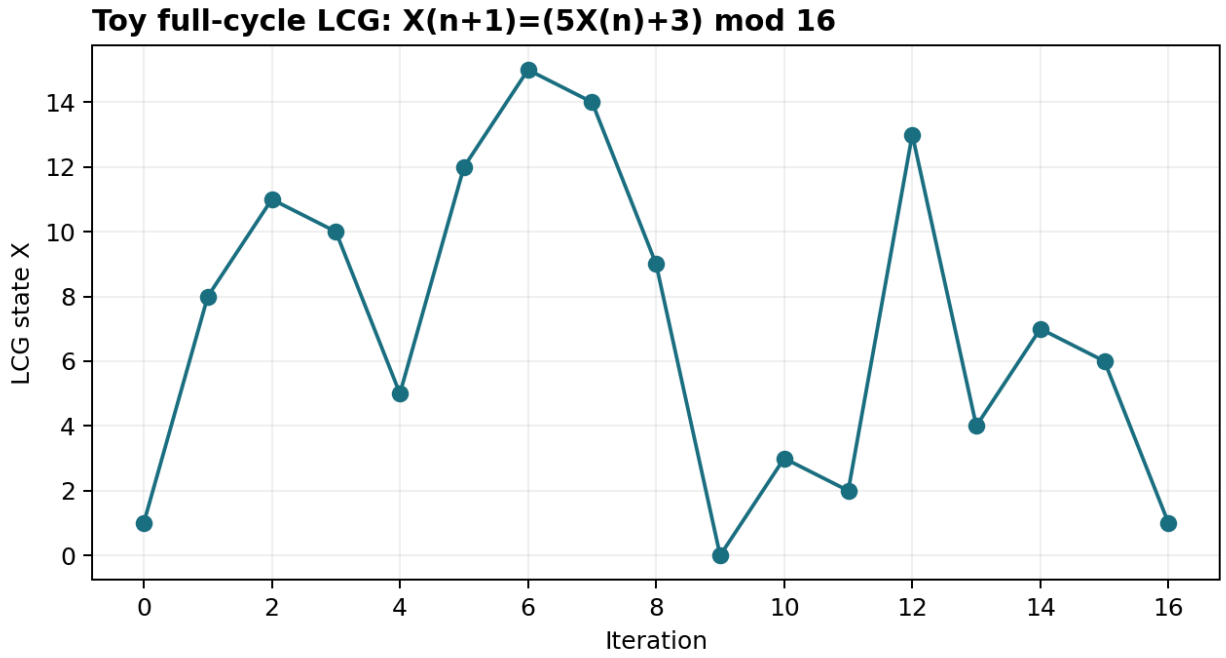

Figure 6. Original complete state trace of the tiny teaching LCG. The cycle length is exactly 16; the generator is deliberately unsuitable for serious simulation due to small state and visible regularity.

9.1 Period and the limitations of tiny generators

Because a PRNG has finitely many states, its state sequence must eventually repeat. Good modern generators use vastly larger state spaces and carefully tested algorithms. Short periods, correlations between consecutive values and dependence on particular parameter choices can bias experiments.

9.2 Modern reproducible simulation practice

In Python, NumPy provides a `Generator` interface to pseudorandom sampling. Record the library version, selected bit generator, seed and mapping. The default implementation is documented by NumPy, but software defaults may evolve; do not assume an experiment will reproduce forever without environment metadata.

Listing 1: Original reproducible example using a seeded NumPy Generator

```
import numpy as np
rng = np.random.default_rng(20261121)
# hypothetical mean 5 minutes, not calibrated to plant data
arrivals = rng.exponential(scale=5.0, size=8)
print(arrivals)
```

Check your understanding
Reproducibility rule: Save the bit-generator name and library version, plus the seed and distribution mapping. For independent simulation replications, use a documented stream-substream procedure, not repeated identical seeds. A seeded single run does not measure output uncertainty.

10 Testing Randomness and Generator Performance

10.1 Uniformity tests

A basic frequency test divides the interval $[0, 1)$ into equal bins and compares the observed counts with equal expected counts. The chi-square mechanics learned earlier apply, but appropriate expected counts and sampling assumptions still matter. A seemingly uniform histogram alone cannot rule out serial dependence.

10.2 Serial correlation and lag plots

Plot U_n versus U_{n+1}. Visible lines or clusters can signal patterns in an unsuitable generator. A lag-correlation statistic can detect some kinds of linear dependence, but a small value does not rule out all nonlinear patterns or higher-dimensional structure.

10.3 Runs tests and periodicity

A runs test examines patterns of consecutive values above or below a threshold. Periodicity tests investigate repetitions and cycle behavior. No finite collection of tests can prove that a deterministic sequence is genuinely independent or

unpredictable. Choose reputable algorithms and evaluate the practical consequences for your simulation outputs.

10.4 Simulation-specific checks

Recompute output measures with multiple controlled seeds and replications. Check for plausible changes in averages, quantiles and rare-event counts. If two simulations unexpectedly track each other, examine whether they reused the same stream or mixed random draws across activities. The measured data and physical model remain equally important.

Two separate quality gates
Input-model quality: Does the selected probability model represent the observed factory process adequately for the decision?
Generator quality: Does the chosen PRNG produce suitable reproducible samples without systematic artifacts for that model?
Passing one gate cannot substitute for the other.

11 Robotics and Manufacturing Applications

11.1 Robot processing and gripper retries

Input model: distribution of pick-and-place durations conditional on product type and retry state. Collect multiple shifts, separate normal cycles and abnormal but genuine retries, then simulate throughput and waiting time under varying demand. A model that deletes all long cycles may understate queue buildup.

11.2 AGV travel and warehouse operations

Input model: travel-time distributions by route length, congestion and charging state. Model the correlation between successive jobs when vehicles share the same congested aisle. Do not assume independently sampled travel times if traffic persists over many minutes.

11.3 Breakdowns and repair

Input model: time-to-failure and repair durations. Differentiate event counts from the duration between events; analyze censored maintenance intervals, preventive service and whether failure mechanisms change after replacements. A homogeneous Poisson model may be a poor choice if equipment ages or undergoes operating-regime changes.

11.4 Supplier and buffer arrivals

Input model: schedules, batch sizes, irregular delays and correlations. Combine order-release timestamps with material availability. A simulation of just a random interarrival stream may miss scheduled, clustered or delayed deliveries.

11.5 End-to-end engineering experiment

1. Define the decision, such as whether a second robot is worth its cost.
2. Gather the relevant arrival, service, breakdown and product-mix logs.
3. Clean and stratify observations; publish the data dictionary and exclusions.
4. Select input models using physical reasoning, plots and appropriate tests.
5. Validate model behavior against independent factory observations.
6. Select a reputable seeded PRNG and document the generator/versions.
7. Run designed replications for alternatives and quantify uncertainty.
8. Recommend an action, state its limits, and monitor results after implementation.

In simple language
The most convincing simulation is not the one with the fanciest curve. It is the one whose input data, assumptions and analysis can be explained and reproduced.

12 Exam Revision Questions and Worked-Answer Checks

1. **What is input modeling?** Choosing and checking a representation for uncertain simulation inputs from observations and engineering knowledge.
2. **Give four data sources.** PLC logs, MES data, instrumented trials and time studies, each with coverage limitations.
3. **Why preprocess?** Correct units and duplicate timestamps, handle missingness, preserve rare true events and document corrections.
4. **Difference between histogram and distribution?** Histogram is a sample summary; a fitted probability distribution is a model requiring justification.
5. **What is an ECDF?** The fraction of observed values not exceeding a given threshold.
6. **Find the example mean.** Twelve times sum to 40 s, so mean $= 3.333$ s.
7. **Example mode and median?** Both 3 s.
8. **State the chi-square GOF statistic.** Sum $(O - E)^2/E$ over suitably defined nonempty bins.
9. **Compute the four-bin statistic.** $O = (9, 11, 10, 10)$ and $E = (10, 10, 10, 10)$ give $\chi^2 = 0.2$.
10. **What is a KS statistic?** The largest gap between sample ECDF and a specified continuous CDF.
11. **Why check autocorrelation?** Correct frequency counts do not guarantee independent time observations.
12. **Compute lag-one example correlation.** For $(2, 2, 3, 3, 4, 4)$ under the stated estimator, $r_1 = 0.5$.
13. **Define PRNG and seed.** A deterministic state algorithm generating reproducible sequences from an initial value/state.
14. **Write the LCG formula.** $X_{n+1} = (aX_n + c)$ mod m.
15. **Find its first three updates.** With $(a, c, m, X_0) = (5, 3, 16, 1)$, outputs are 8, 11, 10.
16. **LCG period for the teaching example?** 16 updates before the seed returns.

17. **Is one frequency test enough to prove randomness?** No; dependence, period and generator suitability also matter.
18. **How to model robot cycle times?** Gather shifts/product classes, inspect distributions and dependencies, validate against held-out data and test sensitivity.
19. **Why record software version?** Generator implementation and defaults can change across software environments.
20. **Is a nonrejected fit a proven true model?** No; GOF nonrejection is not physical validation.

Sources, Figure Provenance and Limits

Primary syllabus: SPPU TE Robotics & Automation 2024 Pattern, *TE Syllabus.pdf*, PEC321-ROA (B), Unit III, physical PDF p.27. This document follows its stated topic list. The connected folder's M&S Units I–II lecture and question-bank PDFs were opened for earlier units; a separate Unit III lecture has not been directly verified.

External references consulted: NIST Engineering Statistics Handbook, Chi-square goodness-of-fit and Kolmogorov–Smirnov goodness-of-fit. NumPy documentation: Random Generator API. Python documentation: Random numbers.

Figure ledger: All included illustrations, plots and workflows were produced for this study guide from explicitly synthetic data. No third-party textbook illustration or copyrighted photograph was reproduced. Prepared PNGs and editable SVGs are included with the source package.

Verification limits: Arithmetic, page structure and text presence can be checked automatically. Neither a statistical GOF test nor a PRNG test proves a model valid for production use. This learning edition does not claim independent academic proofreading or validation against real factory observations.

UNIT III | P0 syllabus-gap corrections

Supplement to the existing reviewed edition. All original teaching pages remain unchanged. The material below closes only the specifically rejected syllabus/question-bank gaps.

1. Departmental Q17: Mixed congruential generator

Given X0 = 27, multiplier a = 17, increment c = 43 and modulus m = 100. Use X(n+1) = (17 X(n) + 43) mod 100, followed by R(n) = X(n)/100. The five requested values are X1 through X5; X0 is the seed, not the first generated number.

n	Calculation	X(n)	R(n)
1	(17(27)+43) mod 100 = 502 mod 100	2	0.02
2	(17(2)+43) mod 100 = 77	77	0.77
3	(17(77)+43) mod 100 = 1352 mod 100	52	0.52
4	(17(52)+43) mod 100 = 927 mod 100	27	0.27
5	(17(27)+43) mod 100 = 502 mod 100	2	0.02

Thus the first five generated pseudorandom fractions are 0.02, 0.77, 0.52, 0.27, 0.02. Since X4 = X0, the state sequence has period 4, demonstrating that these parameters do not give a full-period generator. The fifth value repeats the first.

2. Departmental Q18: Multiplicative congruential generator

With X0 = 3, a = 5, c = 0, m = 16: X(n+1) = 5 X(n) mod 16. Updates: X1 = 15, X2 = 11, X3 = 7, X4 = 3 (= X0). The cycle is 3 → 15 → 11 → 7 → 3; its period is exactly 4, not 16. Normalized generated values are 15/16 = 0.9375, 11/16 = 0.6875, 7/16 = 0.4375, then 3/16 = 0.1875.

3. Departmental Q16 and Q19: Conditions and comparisons

The mixed LCG is X(n+1) = (aX(n)+c) mod m, allowing a nonzero increment. A multiplicative LCG uses c = 0. An additive congruential generator can refer to additive/lagged recurrence, such as X(n) = [X(n-p)+X(n-q)] mod m; specify the recurrence, lag values and seed states because the term is not a single universal parameterization.

For a mixed LCG to have full period m, the Hull–Dobell conditions are: (i) gcd(c,m)=1; (ii) each distinct prime factor of m divides a−1; and (iii) if m is divisible by 4, then a−1 is divisible by 4. They are conditions for full period, not a guarantee of statistical independence. In Q17, gcd(43,100)=1, but a−1=16 is not divisible by 5; therefore full period is impossible.

4. Departmental Q20 and Q21: Frequency tests and uses

Chi-square uniformity test: divide [0,1) into k predeclared equal-width classes; with N draws, each expected count is E=N/k and $\chi^2 = \Sigma(O-E)^2/E$. Check adequate expected counts before using the chi-square approximation. K–S uniformity test: sort N values and compute D+ = max(i/N−U(i)), D− = max(U(i)−(i−1)/N), D = max(D+,D−); use an appropriate critical value for a fully specified continuous Uniform(0,1) null. Neither test establishes independence, period length or cryptographic security.

Simulation applications: transform PRNG draws into arrival intervals for robot queues, demand for inventory replenishment, and process/repair times for production systems. Keep per-replication seeds or generator states, record the sampling algorithms and verify correlations so identical or overlapping streams do not bias results.

5. Departmental Q22: Workcell input-model answer structure

Collect timestamped service-completion records across products, shifts, retries and maintenance states; reconcile clocks/units, identify censoring and missing values; plot a histogram, ECDF, Q–Q comparisons and time-ordered trends; choose candidate empirical/positive-support distributions; estimate parameters; check fit with justified chi-square or K–S methods and inspect autocorrelation; independently validate on held-out data; record model assumptions and repeat sensitivity runs. A fitted distribution is not automatically evidence of physical validity.

Question-bank alignment: M&S Question Bank.pdf, Unit III, Q16–Q22. All numerical inputs above originate from that departmental question bank.