Time-based Simulation Engine (Flanders Make *Costleap* project)

Yue Cai[1,2] Robert Boute[1,2,3] Laurens Deprez[1]

[1]Research Center for Operations Management, KU Leuven, Belgium

[2]Flanders Make@KU Leuven, Leuven, Belgium

[3]Technology and Operations Management Area,
Vlerick Business School, Leuven, Belgium

1 Overview of the Simulation Engine

This simulation engine generates comprehensive failure histories for industrial equipment within predefined observation periods. The simulation engine allows the inclusion of machine-specific risk factors, which can affect failure behavior. For each unit, the simulation produces a sequence of failures, their types, and the corresponding maintenance actions and costs, enabling comprehensive reliability analysis and maintenance optimization studies.

1.1 Input and Output

1.1.1 Input parameters

Basic Simulation Parameters

Parameter	Type	Description
`n_machines`	int	Number of machines to simulate
`t_obs`	int	Machine observation horizon
`m`	int	Number of discrete time intervals for numerical integration
`delta_t`	float	Time step size, calculated as t_{obs}/m

Preventive Maintenance Parameters

Parameter	Type	Description
T_machines	dict	Per-machine PM intervals: {machine_id: T_i}
push	float	PM effectiveness parameter $(0-1)$. Higher values indicate more effective PM in reducing virtual age

Minor Failure Intensity Parameters

Parameter	Type	Description
include_minor	bool	Whether to include minor failures in the model
model_type_minor	str	Hazard model type: "linear", "log-linear", "weibull"
shape_minor	float	Shape parameter for Weibull distribution (if applicable)
scale_minor	float	Scale parameter for hazard function
intercept_minor	float	Intercept parameter (for linear/log-linear models)
with_covariates_minor	bool	Whether minor failure intensity depends on covariates

Catastrophic Failure Intensity Parameters

Parameter	Type	Description
include_catas	bool	Whether to include catastrophic failures in the model
model_type_catas	str	Hazard model type: "linear", "log-linear", "weibull"
shape_catas	float	Shape parameter for Weibull distribution
scale_catas	float	Scale parameter for hazard function
intercept_catas	float	Intercept parameter (for linear/log-linear models)
with_covariates_catas	bool	Whether catastrophic failure intensity depends on covariates

Risk Factors Specifications

Parameter	Type	Description

`fixed_covs`	array	Machine-specific time-invariant covariates with shape $(n_{machines}, n_{fixed})$
`machines_dynamic_covs`	dict	{`machine_id: array`} with per-machine dynamic covariate arrays of shape $(m+1, n_{\text{dynamic}})$
`beta_fixed`	array	Hazard coefficients for fixed covariates
`beta_dynamic`	array	Hazard coefficients for dynamic covariates
`cov_update_fn`	func	Optional callback to update dynamic covariates

Minor Failure Type Classification

Parameter	Type	Description
`n_minor_types`	int	Number of minor subtype categories n
`beta_multinom_fixed`	array	Multinomial logit coefficients for fixed covariates, shape $(n_{fixed}, n_{types} - 1)$
`beta_multinom_dynamic`	array	Multinomial logit coefficients for dynamic covariates, shape $(n_{dynamic}, n_{types} - 1)$

Cost Model (Gamma) and Copula Parameters

Parameter	Type	Description
`use_covariates`	bool	Whether costs use covariates (location parameter built from covariates).
`theta_copula`	dict	Frank copula parameter for paired minor types (if any). Dict may map composite type → parameter.
Catastrophic costs		
`gamma_coeffs_cat_fixed`	array	Fixed-covariate coefficients for catastrophic cost location
`gamma_coeffs_cat_dynamic`	array	Dynamic-covariate coefficients for catastrophic cost location
`shape_cat`, `scale_cat`	float	Gamma shape and scale for catastrophic costs.
`loc_fixed_cat`	float	Location when use_covariates=False
Minor costs (per subtype $i = 1, \ldots, n$)		
`gamma_coeffs_minor` i`_fixed`	array	Fixed-covariate coeffs for minor type i cost location

gamma_coeffs_minor i_dynamic	array	Dynamic-covariate coeffs for minor type i cost location
shape_minor$i$, scale_minor$i$	float	Gamma shape and scale for minor type i costs
loc_fixed_minor$i$	float	Location when use_covariates=False
Preventive Maintenance Cost Parameters		
gamma_coeffs_pm_fixed	array	Fixed-covariate coeffs for PM cost location
gamma_coeffs_pm_dynamic	array	Dynamic-covariate coeffs for PM cost location
shape_pm, scale_pm	float	Gamma shape and scale for PM costs
loc_fixed_pm	float	Location when use_covariates=False

1.1.2 Output Data Structures

The simulation engine returns two primary datasets:

Failure Events Dataset (results_df) *Event Identification*

Column	Type	Description
machine_id	int	Machine identifier (1 to $n_{machines}$)
failure_time	float	Continuous failure time within current cycle
absolute_time	float	Absolute failure time considering catastrophic resets

Failure Classification

Column	Type	Description
risk_type	int	Competing risk type: 1 =minor, 2 =catastrophic, 0 =censored
failure_type	int	Specific failure type: -1 =catastrophic, 0 =censored, 1/2/3 =minor subtypes
censor_status	int	Observation status: 1 =observed failure, 0 =censored

Cost Information

Column	Type	Description

`failure_cost`	float	Repair cost for this failure event
`pm_cost`	float	cost for pm event

Covariate Values at Failure Time

Column	Type	Description
`dynamic_cov_1`, `dynamic_cov_2`, ...	float	Dynamic covariate values
`fixed_cov_1`, `fixed_cov_2`, ...	int/float	Machine-specific fixed covariate values

Dynamic Covariate History Dataset A DataFrame tracking the evolution of dynamic covariates over time:

Column	Type	Description
`machine_id`	int	Machine identifier
`time_point`	float	Continuous time point
`time_index`	int	Discrete time index
`dynamic_cov_1_history`, `dynamic_cov_2_history`, ...	float	Values of each dynamic covariate dimension at each time point

1.2 Examples

Figure 1 illustrates a timeline of failure events for 5 machines, showing how the simulation captures event sequences, types, and timing in a tangible and interpretable manner.

2 Model Assumptions and Model Framework

2.1 Assumptions

- **Conditional independence of failures and costs.** We assume that failure occurrence processes and maintenance cost magnitudes are conditionally independent. This assumption enables a computationally efficient two-stage simulation approach where failure events are first

	machine_id	abs_fail_time	risk_type	failure_type	censor_status	failure_cost	fixed_cov_1	fixed_cov_2	fixed_cov_3	fixed_cov_4	dynamic_cov_1_at_failure
0	1	3.85	2	−1	1	290.320270	1.0	1.0	0.0	0.0	0.0
1	1	5.00	0	0	0	0.000000	1.0	1.0	0.0	0.0	0.0
2	2	1.10	1	2	1	54.713328	0.0	0.0	1.0	0.0	0.0
3	2	1.40	1	3	1	105.527792	0.0	0.0	1.0	0.0	1.0
4	2	3.20	1	2	1	79.773283	0.0	0.0	1.0	0.0	1.0
5	2	5.00	0	0	0	0.000000	0.0	0.0	1.0	0.0	0.0
6	3	0.40	1	2	1	81.611224	1.0	1.0	1.0	1.0	0.0
7	3	1.60	1	3	1	108.190372	1.0	1.0	1.0	1.0	1.0
8	3	3.45	1	3	1	94.461289	1.0	1.0	1.0	1.0	1.0
9	3	4.40	1	1	1	332.839026	1.0	1.0	1.0	1.0	1.0
10	3	4.80	1	2	1	100.078681	1.0	1.0	1.0	1.0	1.0
11	3	5.00	0	0	0	0.000000	1.0	1.0	1.0	1.0	0.0
12	4	3.50	1	2	1	75.570066	1.0	1.0	0.0	0.0	0.0
13	4	4.15	1	1	1	123.322372	1.0	1.0	0.0	0.0	0.0
14	4	4.30	1	2	1	104.331540	1.0	1.0	0.0	0.0	0.0
15	4	4.50	2	−1	1	232.700586	1.0	1.0	0.0	0.0	0.0
16	4	5.00	0	0	0	0.000000	1.0	1.0	0.0	0.0	0.0
17	5	1.35	1	1	1	172.572559	1.0	0.0	1.0	1.0	0.0
18	5	2.40	2	−1	1	399.577379	1.0	0.0	1.0	1.0	0.0
19	5	3.90	1	1	1	133.664009	1.0	0.0	1.0	1.0	0.0
20	5	4.45	1	1	1	135.517635	1.0	0.0	1.0	1.0	0.0
21	5	4.50	1	1	1	192.408845	1.0	0.0	1.0	1.0	0.0
22	5	4.95	1	2	1	174.355501	1.0	0.0	1.0	1.0	0.0
23	5	5.00	0	0	0	0.000000	1.0	0.0	1.0	1.0	0.0

Figure 1: Illustrative timeline of simulated failure events for 5 machines.

generated according to underlying stochastic processes, followed by cost generation.

- **Exogeneity of preventive maintenance (PM).** The PM schedule is assumed to be exogenously given and does not depend on the real-time system state. While PM actions may reduce the intensity of future failures by partially restoring system condition, their timing is predetermined and not adaptively optimized in the failure generation process.

2.2 System Setup and Observation Period

Consider a system observed over a finite time horizon $[0, t_{obs}]$ with n preventive maintenance (PM) actions scheduled at regular intervals. The PM interval is defined as:

$$T = \frac{t_{obs}}{n+1}$$

The observation time t_{obs} for each machine can follow either:

- Fixed observation: $t_{obs} = t_{max}$ (constant across all machines)
- Random observation: $t_{obs} \sim \text{Uniform}(t_{min}, t_{max})$

A Bernoulli trial with probability p determines the observation mode for each machine.

2.3 Time Discretization

To handle time-varying covariates and enable efficient computation of cumulative intensity functions, the continuous observation period $[0, t_{obs}]$ is discretized into m equal intervals:

$$\Delta t = \frac{t_{obs}}{m}, \quad t_k = k \cdot \Delta t, \quad k = 0, 1, \ldots, m$$

Discretization facilitates numerical approximation of cumulative hazards, allows dynamic covariates to be updated at regular intervals, reduces computational burden compared to continuous-time simulation, and simplifies implementation.

2.4 Covariate Structure

The simulation engine allows the inclusion of machine-specific risk factors to account for heterogeneity arising from differences in operating conditions, maintenance quality, or inherent characteristics. The risk factors (covariates) can be both fixed and dynamic.

Fixed covariates. Fixed covariates represent static characteristics of each machine that do not change over time, such as design attributes, country of operation, or general operating environment. For each machine i, a vector of fixed covariates is denoted as $\mathbf{x}_i^{\text{fixed}}$.

Dynamic covariates. Dynamic covariates capture time-dependent operational conditions or indicators that evolve during the observation period, for example, cumulative usage, environmental stress, or failure history. They are represented as a matrix $\mathbf{X}^{\text{dynamic}} \in \mathbb{R}^{m+1 \times r}$, where $m+1$ is the number of discrete time points and r is the number of dynamic features.

3 Failure Process Modeling

3.1 Competing Risks

Two competing failure processes can be modeled in this system:

- **Minor Failures**: Degradation-driven failures that result from cumulative wear or material aging. They are generally non-fatal and can be mitigated or postponed through preventive maintenance (PM) actions.

- **Catastrophic Failures**: Fatal shock processes independent of internal degradation. Preventive maintenance does not directly affect their arrival rate.

Let $\lambda_m(t)$ and $\lambda_c(t)$ denote the intensity functions for minor and catastrophic failures, respectively. The total system intensity is:

$$\lambda_{total}(t) = \lambda_m(t) + \lambda_c(t)$$

3.2 Maintenance actions

Two types of maintenance actions are implemented: *preventive maintenance (PM)* and *corrective maintenance (CM)*.

Preventive Maintenance (PM). PM is carried out before a system failure. Its effectiveness is measured by the *restoration factor* $k_{\lambda_0} \in [0, 1]$. Here, $k_{\lambda_0} = 1$ corresponds to "as good as new" (perfect repair), $k_{\lambda_0} = 0$ to "as bad as old" (minimal repair), and $0 < k_{\lambda_0} < 1$ to imperfect repair. In our current implementation, we assume that the restoration factor is constant across all PM actions, although it can be adjusted to vary by action or machine if desired.

When failure occurrences are affected by the PM scheme (e.g., minor failures), we adopt the concept of *virtual age* to capture the impact. Virtual age is a theoretical age that reflects the accumulated degradation of the system considering maintenance effects, rather than the chronological time. Specifically, after each PM action, the virtual age is reset according to

$$V(t) = (t \bmod T) + (1 - k_{\lambda_0}) \cdot T \cdot \lfloor t/T \rfloor,$$

where T is the PM interval, $t \bmod T$ is the time since the last PM, and $\lfloor t/T \rfloor$ counts the number of completed PM cycles.

When failure occurrences are not affected by the PM scheme (e.g., catastrophic failures), the PM actions do not influence the hazard rate, and the virtual age is equal to the actual elapsed time:

$$V(t) = t,$$

Corrective Maintenance (CM). Corrective maintenance (CM) is performed immediately after a system failure and restores the system state depending on the type of failure:

- **CM for Minor Failures:** For minor failures, we assume a minimal corrective maintenance (as-good-as-old) action, which does not significantly alter the system's hazard.
- **CM for Catastrophic Failures:** Catastrophic failures require system replacement or overhaul. After a catastrophic failure, the system is restored to an as-good-as-new state. All states reset.

3.3 Failure intensity

3.3.1 Baseline Intensity Functions

The engine supports multiple baseline intensity specifications:

Linear Model:

$$\lambda_0(V(t)) = \alpha + \beta \cdot V(t)$$

Log-Linear Model:

$$\lambda_0(V(t)) = \exp(\alpha + \beta \cdot V(t))$$

Weibull Model:

$$\lambda_0(V(t)) = \frac{\eta}{\theta}\left(\frac{V(t)}{\theta}\right)^{\eta-1}$$

where η is the shape parameter and θ is the scale parameter.

3.3.2 Covariate Effects

Covariate effects are incorporated through a proportional hazards specification when they are included in the model. Specifically, if fixed and/or dynamic covariates are available, the hazard function is modified as

$$\lambda(t \mid \mathbf{x}_{\text{fixed}}, \mathbf{x}_{\text{dynamic}}) = \lambda_0(t) \cdot \exp\left(\mathbf{x}_{\text{fixed}}^{\top}\boldsymbol{\beta}_{\text{fixed}} + \mathbf{x}_{\text{dynamic}}^{\top}\boldsymbol{\beta}_{\text{dynamic}}\right),$$

where $\lambda_0(t)$ is the baseline hazard (which may depend on virtual age if PM affects the process), $\mathbf{x}_{\text{fixed}}$ and $\mathbf{x}_{\text{dynamic}}$ are vectors of fixed and dynamic covariates, and $\boldsymbol{\beta}_{\text{fixed}}$, $\boldsymbol{\beta}_{\text{dynamic}}$ are their respective coefficients. If covariates are not included, the hazard reduces to the baseline form $\lambda(t) = \lambda_0(t)$.

3.3.3 Cumulative Intensity Computation

For computational efficiency, cumulative intensities are computed incrementally:

Closed-form solutions (catastrophic failures without covariates):

$$\Lambda(t) = \begin{cases} \alpha t + \frac{\beta t^2}{2} & \text{(Linear)} \\ \frac{e^{\alpha}}{\beta}(e^{\beta t} - 1) & \text{(Log-linear)} \\ \left(\frac{t}{\theta}\right)^{\eta} & \text{(Weibull)} \end{cases}$$

Numerical integration (cases with covariates or PM effects):

$$\Lambda(a, b) = \int_a^b \lambda(s) ds$$

3.4 Failure Arrival Modeling with NHPP

The framework models failure arrivals using a *non-homogeneous Poisson process* (NHPP). In this setup, the NHPP is characterized by its time-dependent intensity (hazard) function $\lambda(t)$ and the corresponding cumulative intensity

$$\Lambda(t) = \int_0^t \lambda(u)\, du.$$

Inverse Transform Sampling of NHPP Failure Times To simulate failure arrivals from a non-homogeneous Poisson process (NHPP) with cumulative intensity $\Lambda(t)$, we use the inverse transform method. The procedure is as follows:

1. Generate a uniform random variable $U \sim \text{Uniform}(0, 1)$.
2. Transform U to a standard exponential variable:
$$s = -\log(U) \sim \text{Exp}(1).$$
3. Determine the next failure time t_f by solving
$$\Lambda(t_f) = s.$$

If $\Lambda(t)$ has a closed-form expression, this can be solved analytically. Otherwise, numerical integration and root-finding are used, especially when $\lambda(t)$ depends on virtual age, dynamic covariates, or maintenance actions. This method ensures that the simulated failure times correctly follow the time-varying hazard structure of the NHPP.

3.5 Failure Type Classification

3.5.1 Failure Type Determination via Competing Risks

At each simulated failure time t^*, the system may experience either a minor or a catastrophic failure. These two types of failures are modeled as *competing risks*, with hazard functions $\lambda_m(t)$ and $\lambda_c(t)$, respectively.

The probability that a failure occurring at time t^* is of the minor type is given by

$$P(\text{Minor} \mid t^*) = \frac{\lambda_m(t^*)}{\lambda_m(t^*) + \lambda_c(t^*)}.$$

Operationally, once a failure time t^* is generated, a Bernoulli trial with success probability $P(\text{Minor} \mid t^*)$ is performed to assign the failure type.

3.5.2 Minor Failure Subtypes

If the failure is minor, further classification into subtypes can be performed using a multinomial logistic regression model conditional on the fixed and dynamic covariates.

Assuming the J-th subtype is chosen as the reference category, the model specifies

$$\log \frac{P(Y = j \mid \mathbf{x}(t^*))}{P(Y = J \mid \mathbf{x}(t^*))} = \mathbf{x}(t^*)^\top \boldsymbol{\beta}_j, \quad j = 1, \dots, J-1,$$

where $\boldsymbol{\beta}_j$ is the coefficient vector for subtype j relative to the reference category, and $\mathbf{x}(t^*)$ denotes the feature vector at the failure occurrence time t^*.

The probability for each subtype is then

$$P(Y = j \mid \mathbf{x}(t^*)) = \begin{cases} \dfrac{\exp(\mathbf{x}(t^*)^\top \boldsymbol{\beta}_j)}{1 + \sum_{l=1}^{J-1} \exp(\mathbf{x}(t^*)^\top \boldsymbol{\beta}_l)}, & j = 1, \dots, J-1, \\ \dfrac{1}{1 + \sum_{l=1}^{J-1} \exp(\mathbf{x}(t^*)^\top \boldsymbol{\beta}_l)}, & j = J. \end{cases}$$

The feature vector $\mathbf{x}(t^*)$ can include both fixed covariates (static machine attributes) and dynamic covariates, which are evaluated at the failure time t^*.

4 Cost Modeling

Gamma-distributed costs. The maintenance cost is modeled using Gamma distributions, with their location parameters influenced by both fixed and dynamic covariates. For a failure of type k at time t^*, the linear predictor is defined as

$$\ell_k = \mathbf{x}_{\text{fixed}}^\top \boldsymbol{\gamma}_{k,\text{fixed}} + \mathbf{x}_{\text{dynamic}}(t^*)^\top \boldsymbol{\gamma}_{k,\text{dyn}},$$

where $\mathbf{x}_{\text{fixed}}$ denotes time-invariant covariates and $\mathbf{x}_{\text{dyn}}(t^*)$ are time-varying covariates at failure time t^*. The coefficients $\boldsymbol{\gamma}_{k,\text{fixed}}$ and $\boldsymbol{\gamma}_{k,\text{dyn}}$ are type-specific parameters.

Conditional on $\ell_{(k)}$, the failure cost is generated from a Gamma distribution:

$$C_k \sim \text{Gamma}\big(a_k,\ \text{loc} = \ell_k,\ \text{scale} = b_k\big),$$

where a_k is the type-specific scale parameter and b_k is the type-specific scale parameter. Catastrophic and minor failures share this functional form but differ in their parameter values.

Simultaneous failures. When two minor failures occur simultaneously, dependence between their repair costs is introduced via the Frank copula with the cumulative distribution function:

$$C_\theta^{\text{Frank}}(U_1, U_2) = -\frac{1}{\theta} \ln \left(1 + \frac{(e^{-\theta U_1} - 1)(e^{-\theta U_2} - 1)}{e^{-\theta} - 1} \right),$$

where θ controls the strength and direction of dependence.

The conditional sampling scheme proceeds as follows:

1. Let $u_1 = U_1 \sim \text{Uniform}(0, 1)$.
2. Let $V \sim \text{Uniform}(0, 1)$ independently.
3. Compute

$$u_2 = -\frac{1}{\theta} \ln \left(1 + \frac{\big(1 - e^{-\theta}\big)\, V}{e^{-\theta u_1} - V e^{-\theta u_1}} \right), \quad V \sim \text{Uniform}(0, 1).$$

The pair (U_1, U_2) is then transformed into Gamma-distributed costs via the inverse CDF of the corresponding marginals:

$$C_k = F^{-1}_{\text{Gamma},k}\big(U_k \mid a_k, b_k, \ell_k\big), \quad k = 1, 2.$$

The total cost is the sum of the two components:

$$C = C_1 + C_2.$$

5 Running Example

5.1 Covariates Generation

Fixed covariates. For each machine i, a vector of p binary fixed covariates $\mathbf{x}_i^{\text{fixed}}$ is generated as

$$x_{i,j}^{\text{fixed}} \sim \text{Bernoulli}(q_j), \quad j = 1, \dots, p,$$

where q_j is the pre-specified probability for covariate j.

Dynamic covariates. In this example, we design a specific dynamic covariate that captures the occurrence history of Type 3 minor failures.

This binary indicator variable is defined as:

$$X_{dynamic}^{(1)}(t) = \begin{cases} 0 & \text{if no Type 3 minor failure has occurred before time } t \\ 1 & \text{if at least one Type 3 minor failure has occurred before time } t \end{cases}$$

5.2 Specific values for parameters

The complete list of parameter settings used in our demonstration can be found directly in the example script: run_tbm_example.py.