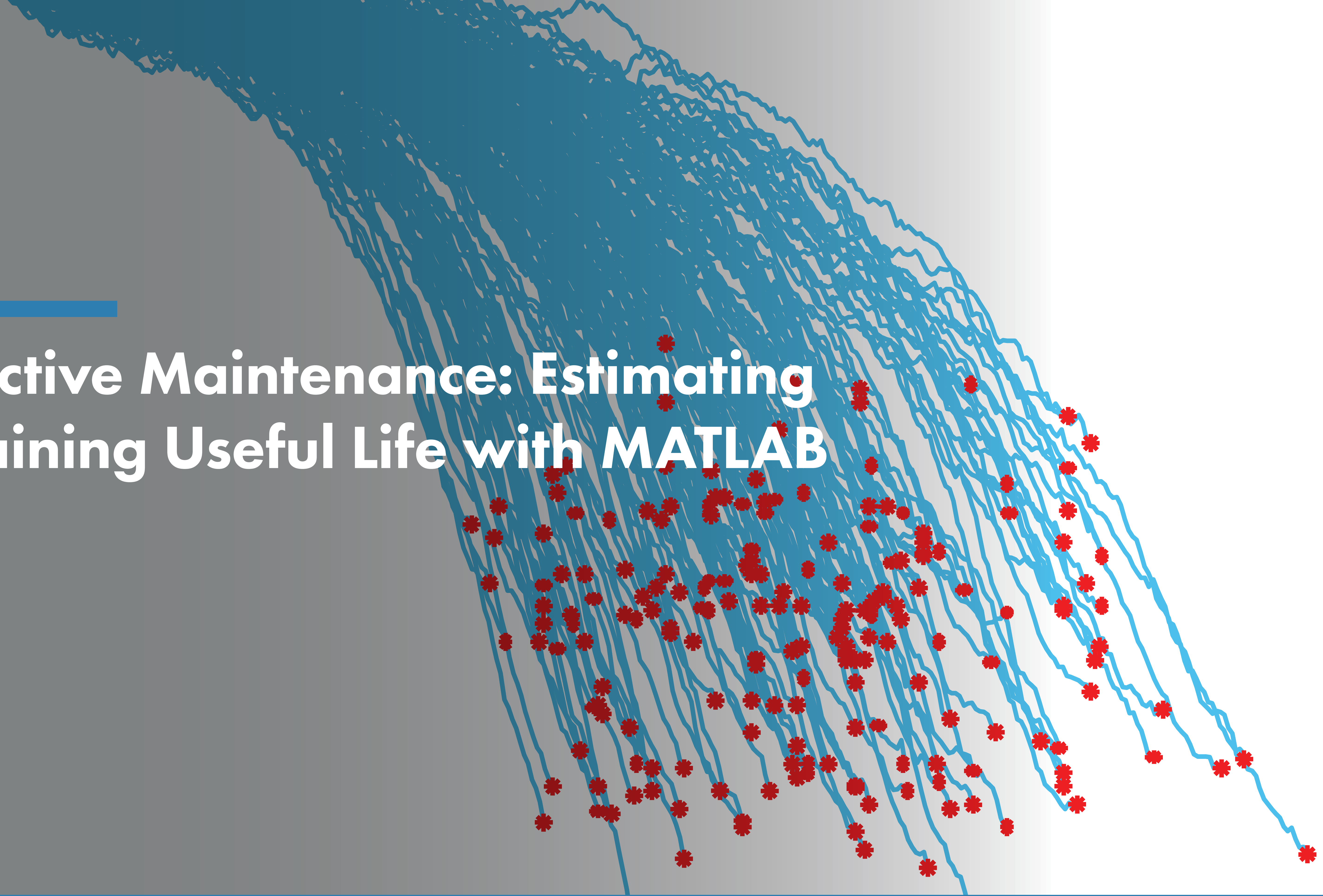
Predictive Maintenance: Estimating Remaining Useful Life with MATLAB



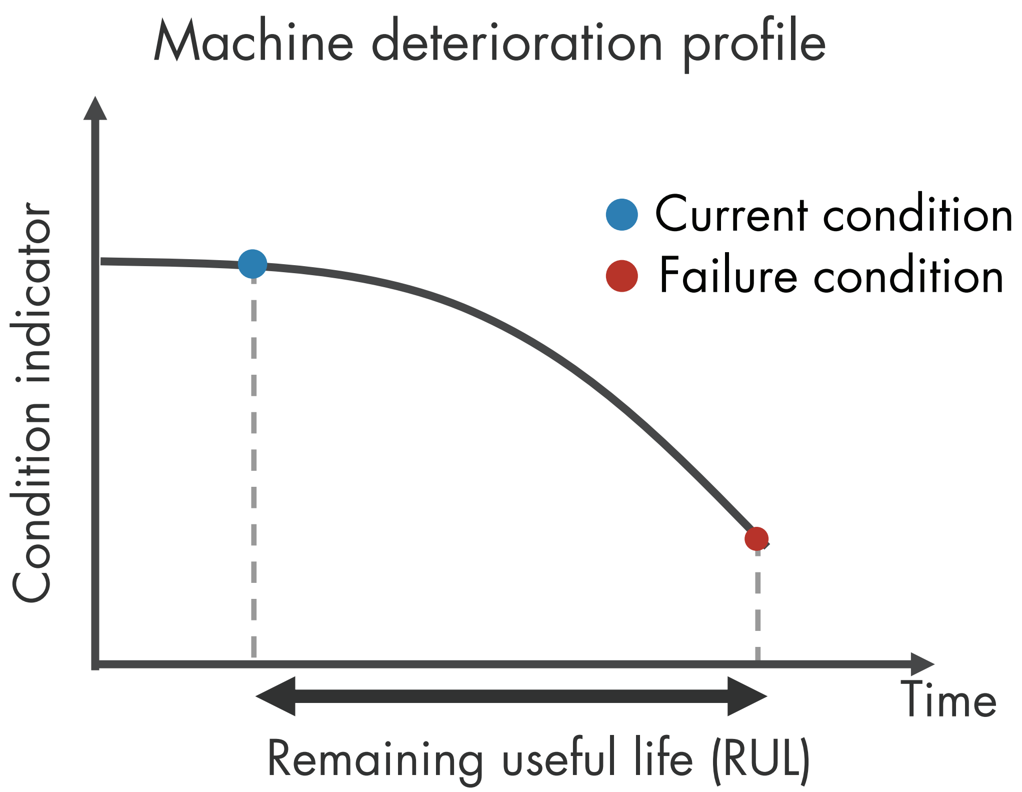
MathWorks®

What Is Remaining Useful Life?

One of the goals of predictive maintenance is to estimate the remaining useful life (RUL) of a system. RUL is the time between a system's current condition and failure. Depending on your system, time can be represented in terms of days, flights, cycles, or any other quantity.

On the plot, we see the deterioration path of a machine over time.

This ebook explores three common models used to estimate RUL (similarity, survival, and degradation) and then walks through the RUL workflow with an example using a similarity model.



Three Common Ways to Estimate RUL

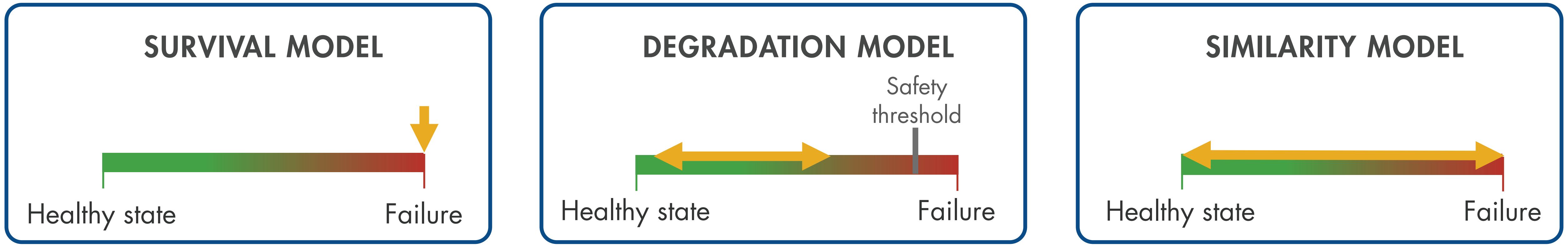
There are three common models used to estimate RUL: similarity, survival, and degradation. Which model should you use? It depends on how much information you have.

Use a *survival model* if you have data only from time of failure rather than complete run-to-failure histories.

Use a *degradation model* if you have some data between a heathy state and failure, and you know a safety threshold that shouldn't be exceeded.

Use a *similarity model* if your data spans the full degradation of a system from a healthy state to failure.

RUL ESTIMATOR MODELS



SURVIVAL MODEL

Healthy state

Failure

DEGRADATION MODEL

Safety threshold

Healthy state

Failure

SIMILARITY MODEL

Healthy state

Failure

Find more detailed information on *RUL Estimator Models* used in predictive maintenance.

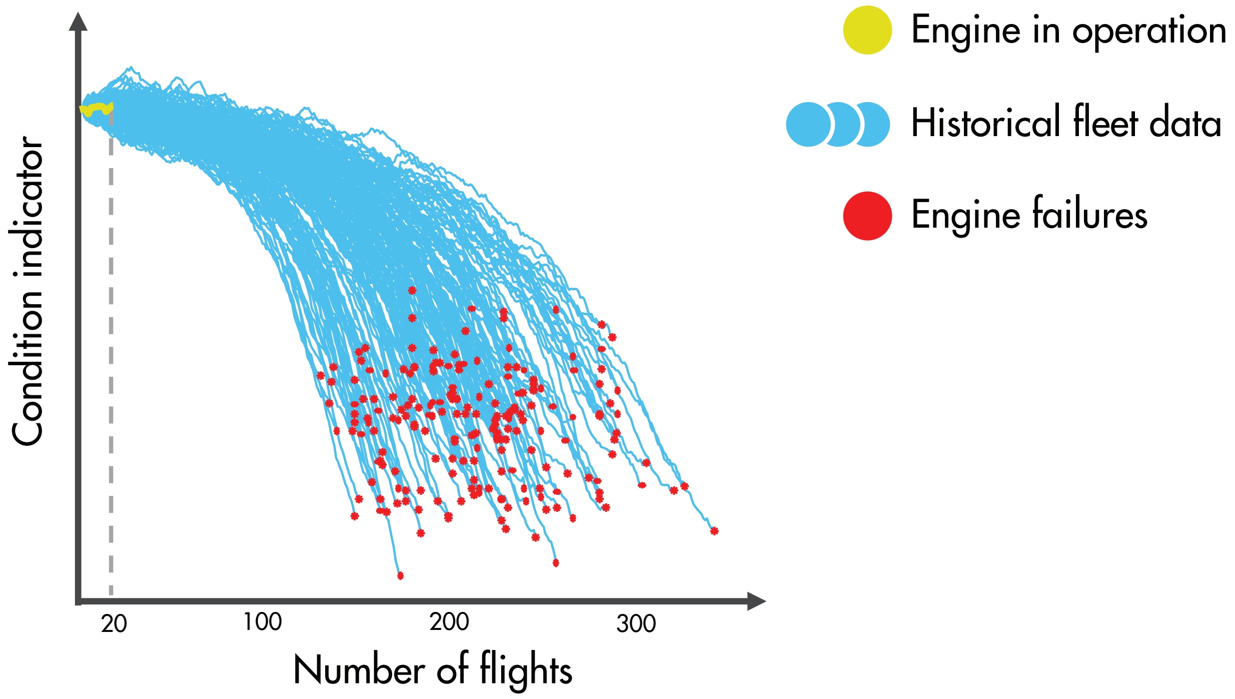
The following aircraft engine example illustrates how estimator models work.

The Working Principle of RUL Estimator Models: The Survival Model

RUL ESTIMATOR MODELS

SURVIVAL MODEL | **DEGRADATION MODEL** | **SIMILARITY MODEL**

Use when you have failure data from similar machines



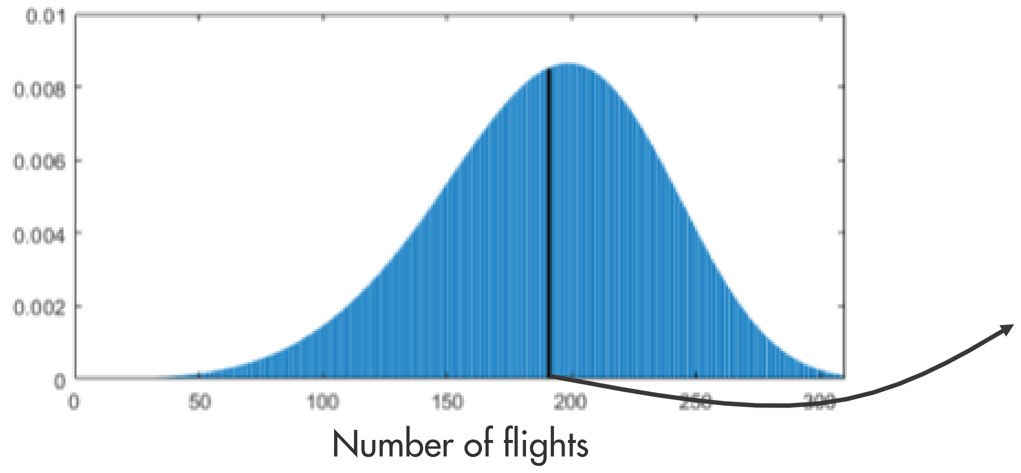
In this example, you want to figure out how many flights the engine can operate until its parts need repair or replacement.

The yellow line on the plot represents your engine, which has been in operation for 20 flights. The blue lines represent the data from a fleet with the same type of engine. The red marks indicate when the engines failed.

If you don't have the complete histories from the fleet, but you do have the failure data, then you can use survival models to estimate RUL.

You can determine how many engines failed after a certain number of cycles (number of flights), and you also know how many flights the engine has been in operation. The survival model uses a probability distribution of this data to estimate the remaining useful life.

Probability density function for the RUL estimation after 20 flights



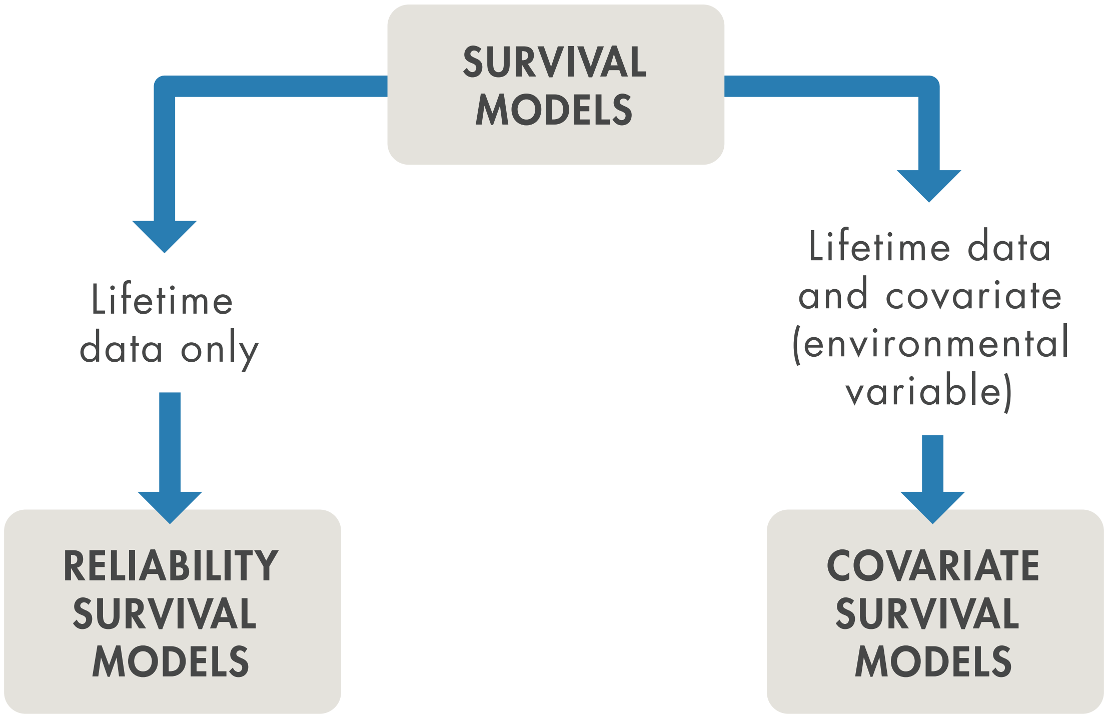
Estimated RUL: **192.3**

The Working Principle of RUL Estimator Models: The Survival Model in MATLAB

Survival analysis is a statistical method used to model time-to-event data. It is useful when you do not have complete run-to-failure histories, but instead have:

Only data about the life span of similar components. For example, you might know how many miles each engine in your ensemble ran before needing maintenance. In this case, you use the MATLAB® model `reliabilitySurvivalModel`. Given the historical information on failure times of a fleet of similar components, this model estimates the probability distribution of the failure times. The distribution is used to estimate the RUL of the test component.

Both life spans and some other variable data (covariates) that correlates with the RUL. Covariates include information such as the component provider, regimes in which the component was used, or manufacturing batch. In this case, use the MATLAB model `covariateSurvivalModel`. This model is a proportional hazard survival model that uses the life spans and covariates to compute the survival probability of a test component.

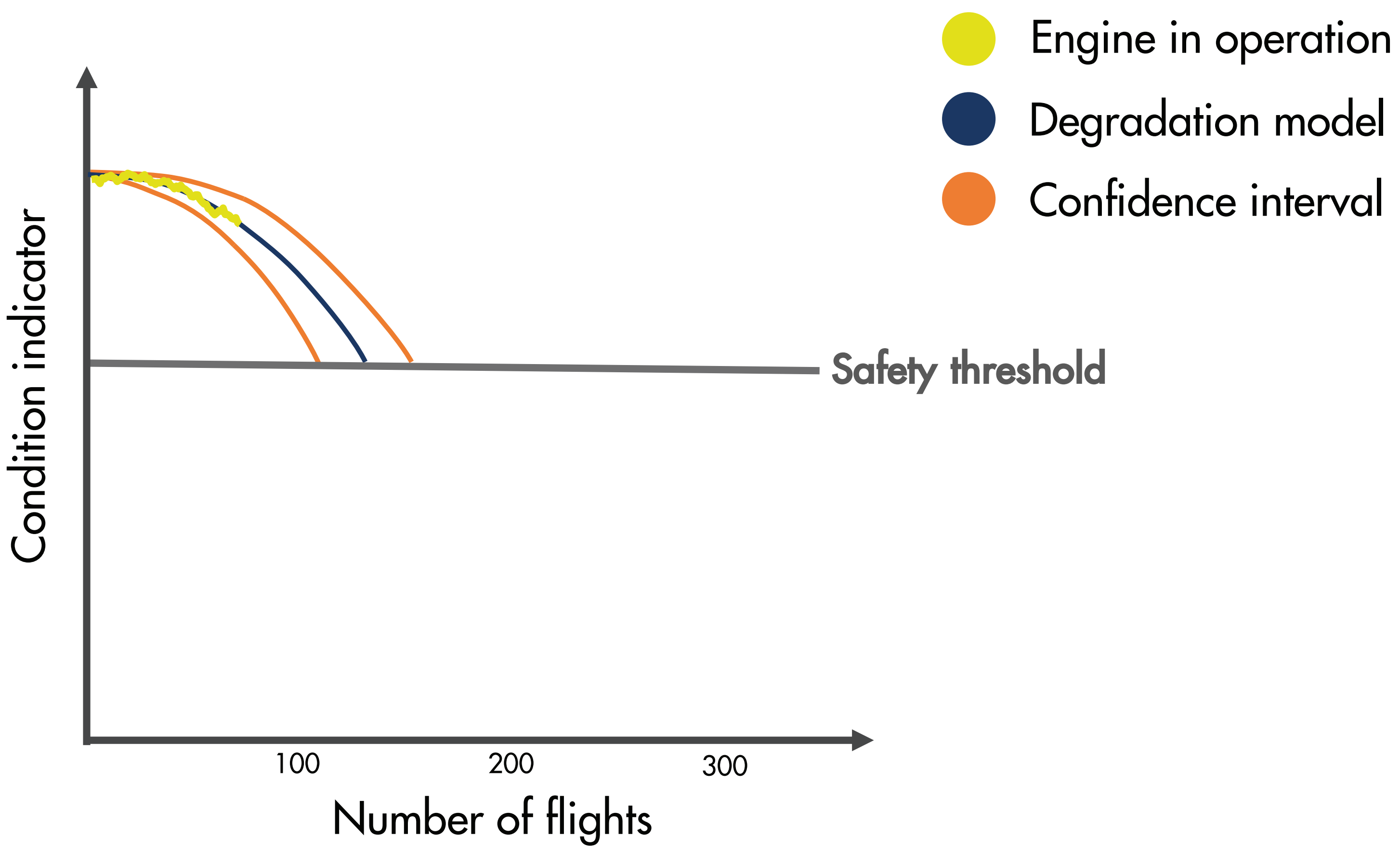

SURVIVAL MODELS
Lifetime data only
Lifetime data and covariate (environmental variable)
RELIABILITY SURVIVAL MODELS
COVARIATE SURVIVAL MODELS

The Working Principle of RUL Estimator Models: The Degradation Model

RUL ESTIMATOR MODELS

SURVIVAL MODEL | DEGRADATION MODEL | SIMILARITY MODEL

Use when you have know a threshold of
some condition indicator that indicates failure



In some cases, no failure data is available from similar machines. But you may have knowledge about a safety threshold that shouldn’t be crossed as this may cause failure. You can use this information to fit a degradation model to the condition indicator, which uses the past information from our engine to predict how the condition indicator will change in the future. This way, you can statistically estimate how many cycles there are until the condition indicator crosses the threshold, which helps you estimate the remaining useful life.

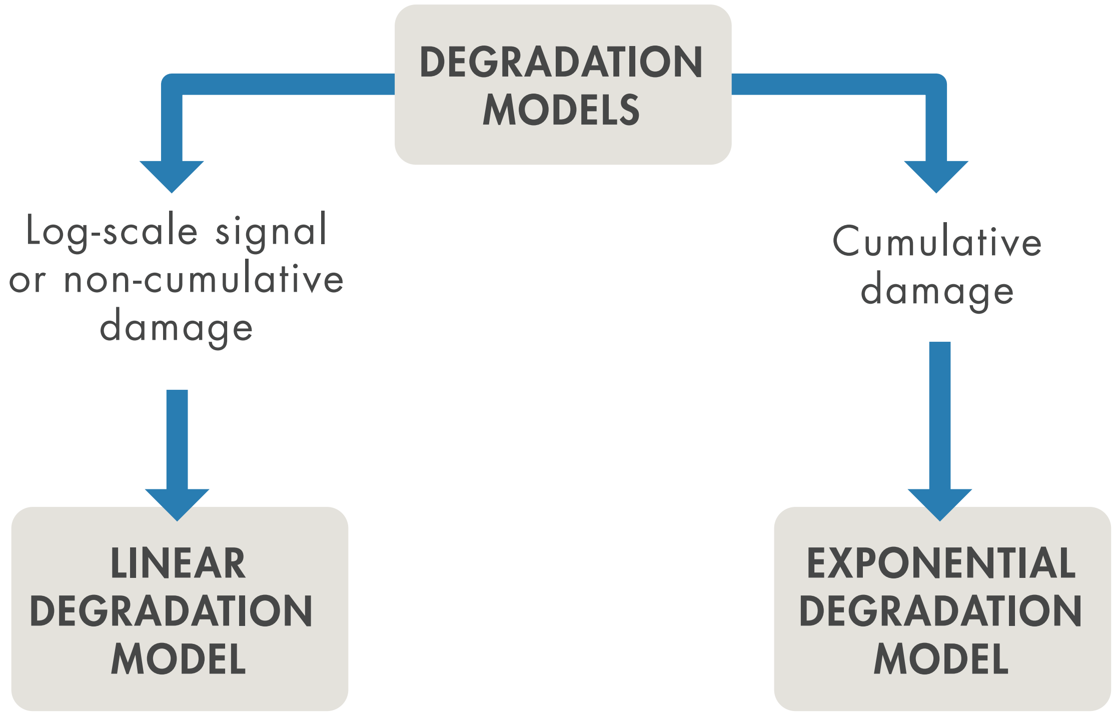
The Working Principle of RUL Estimator Models: The Degradation Model in MATLAB

Degradation models estimate RUL by predicting when the condition indicator will cross a prescribed threshold. These models are most useful when there is a known value of your condition indicator that indicates failure. The two available degradation model types in Predictive Maintenance Toolbox™ are:

Linear degradation model (`linearDegradationModel`) describes the degradation behavior as a linear stochastic process with an offset term. Linear degradation models are useful when your system does not experience cumulative degradation.

Exponential degradation model (`exponentialDegradationModel`) describes the degradation behavior as an exponential stochastic process with an offset term. Exponential degradation models are useful when the test component experiences cumulative degradation.

Degradation models work with a single condition indicator. However, you can use principal-component analysis or other fusion techniques to generate a fused condition indicator that incorporates information from more than one condition indicator.


DEGRADATION MODELS

Log-scale signal or non-cumulative damage

Cumulative damage

LINEAR DEGRADATION MODEL

EXPONENTIAL DEGRADATION MODEL

The Working Principle of RUL Estimator Models: The Similarity Model

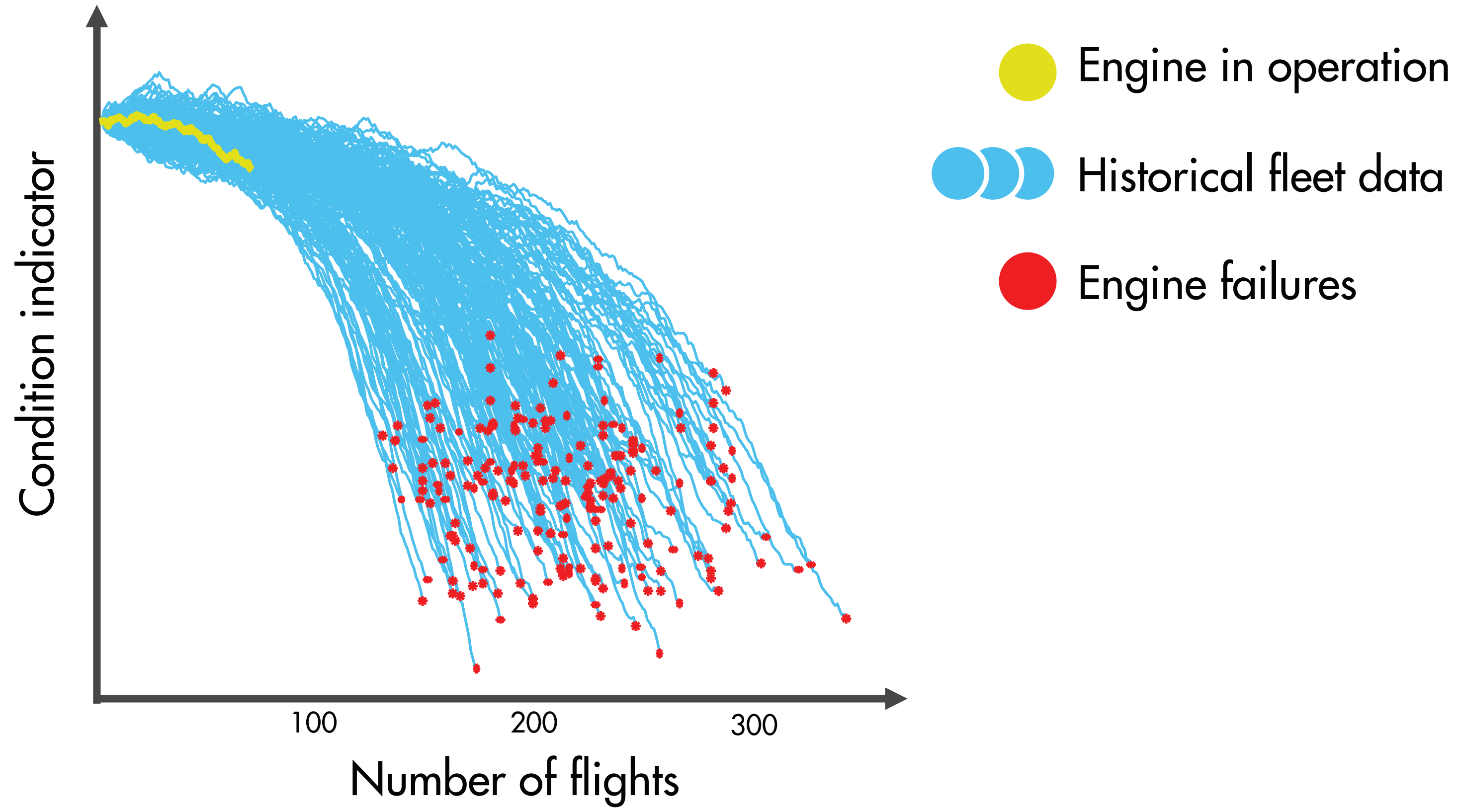
RUL ESTIMATOR MODELS

SURVIVAL MODEL

DEGRADATION MODEL

SIMILARITY MODEL

Use when you have run-to-failure histories from similar machines



Similarity models are useful when you have run-to-failure data (the complete histories from a fleet with the same type of engine, from healthy state through to degradation and failure).

The Working Principle of RUL Estimator Models: The Similarity Model in MATLAB

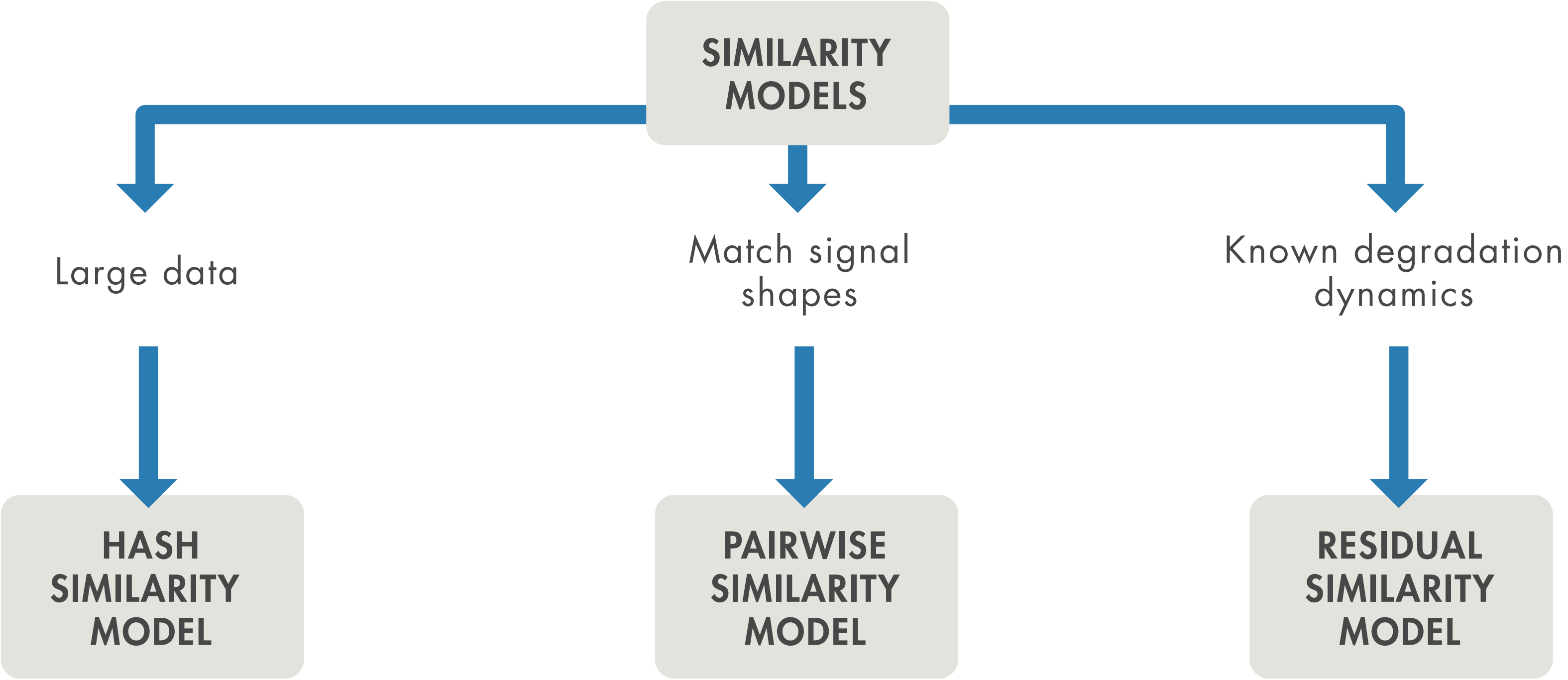
Predictive Maintenance Toolbox includes three types of similarity models:

Hashed-feature similarity model (`hashSimilarityModel`) transforms historical degradation data from each member of your ensemble into fixed-size, condensed information such as the mean, maximum, or minimum values, etc.

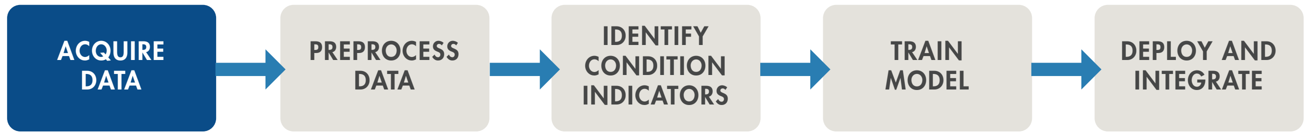
Pairwise similarity model (`pairwiseSimilarityModel`) finds the components whose historical degradation paths are most correlated to those of the test component.

Residual similarity model (`residualSimilarityModel`) fits prior data to a model such as an ARMA model or a model that is linear or exponential in usage time. It then computes the residuals between the data predicted from the ensemble models and the data from the test component. See *Similarity-Based Remaining Useful Life Estimation* for more information.

The following section discusses the similarity model in more detail to illustrate how an RUL prediction is performed.


SIMILARITY MODELS
Large data
Match signal shapes
Known degradation dynamics
HASH SIMILARITY MODEL
PAIRWISE SIMILARITY MODEL
RESIDUAL SIMILARITY MODEL

RUL Estimation Workflow Using the Similarity Model

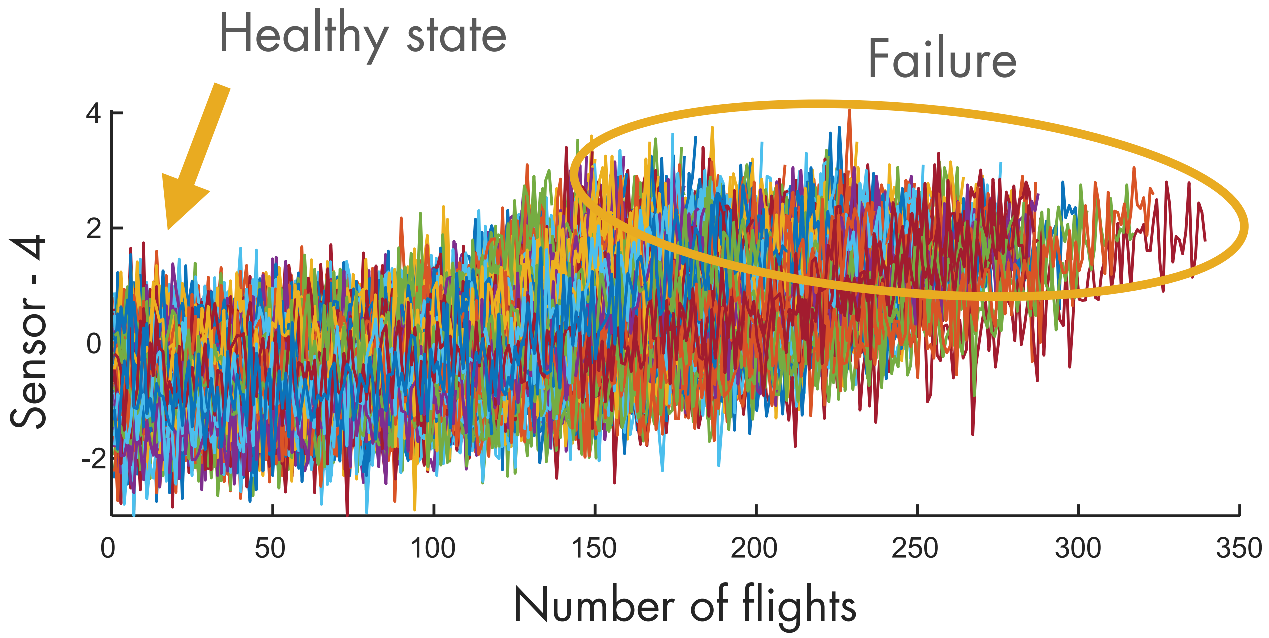


Acquiring Data

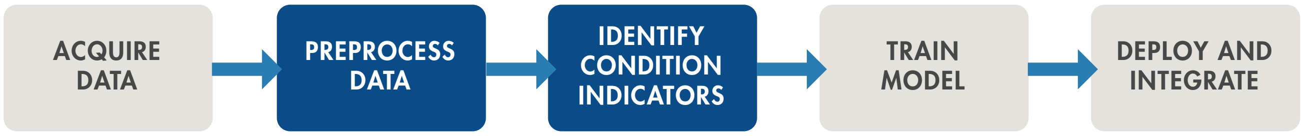
The similarity model is a useful RUL estimation technique. See how this model is used in an example to better understand how an RUL prediction is performed.

The first step when developing a predictive maintenance model is to acquire data. This example uses the Prognostics and Health Management challenge dataset publicly available on *NASA's data repository*. This data set includes run-to-failure data from 218 engines, where each engine dataset contains measurements from 21 sensors. Measurements such as fuel flow, temperature, and pressure are gathered through sensors placed in various locations in the engine to provide measurements to the control system and monitor the engine's health. The plot shows what one sensor's measurements look like for all 218 engines.

On the plot, the x-axis shows the number of cycles (flights), and the y-values represent the averaged sensor values at each flight. Each engine starts in a healthy state and ends in failure.

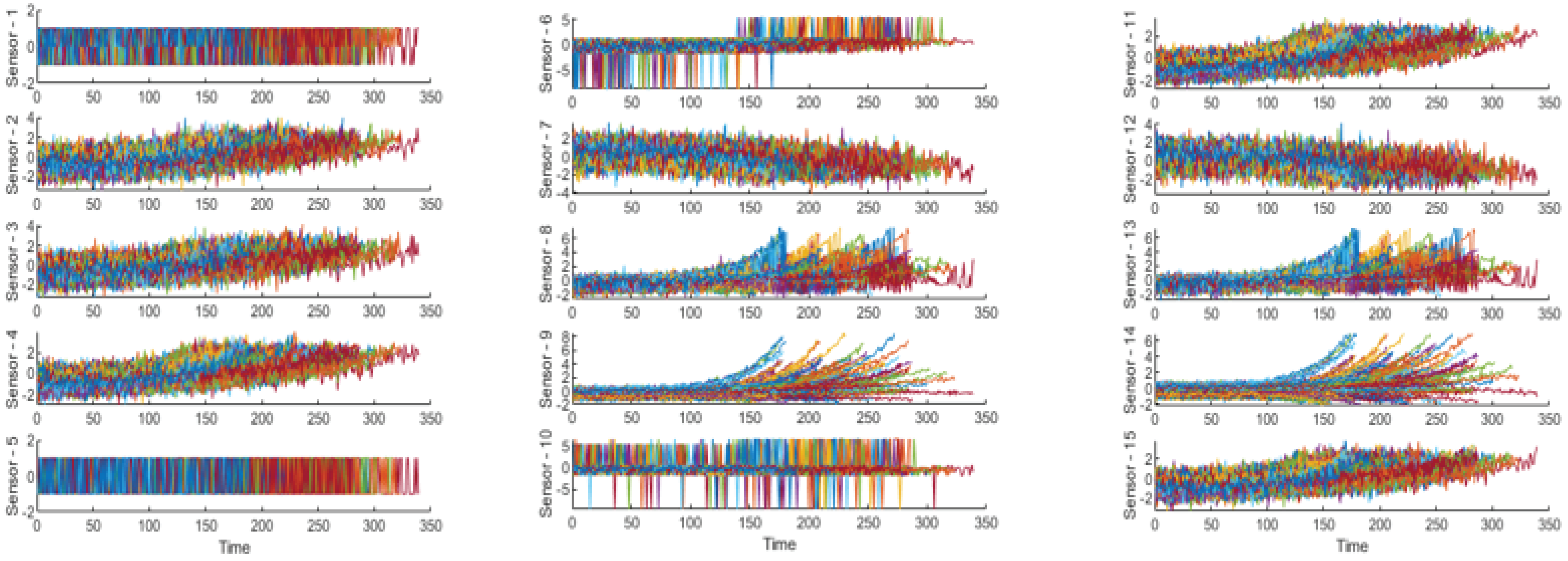


RUL Estimation Workflow Using the Similarity Model *continued*



Preprocessing Data and Identifying Condition Indicators

The previous page showed data from Sensor 4, but the full dataset also includes data from 20 other sensors. If you take a closer look at some of the other sensor readings, you can see that some of these measurements don't show a significant trend toward failure (such as Sensors 1, 5, 6, and 10). Therefore, they won't contribute to the selection of useful features for training a similarity model.



RUL Estimation Workflow Using the Similarity Model *continued*

Preprocessing Data and Identifying Condition Indicators

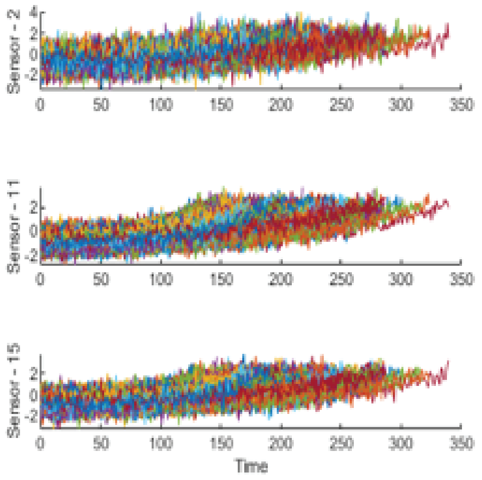
Instead of using all the sensor measurements, identify the three most trendable datasets (the ones that show a significant change in their profile between the healthy state and failure). Sensors 2, 11, and 15 are good candidates to use together to create degradation profiles.

Degradation profiles represent the evolution of one or more condition indicators for each machine in the ensemble (each component), as the machine transitions from a healthy state to a faulty state.

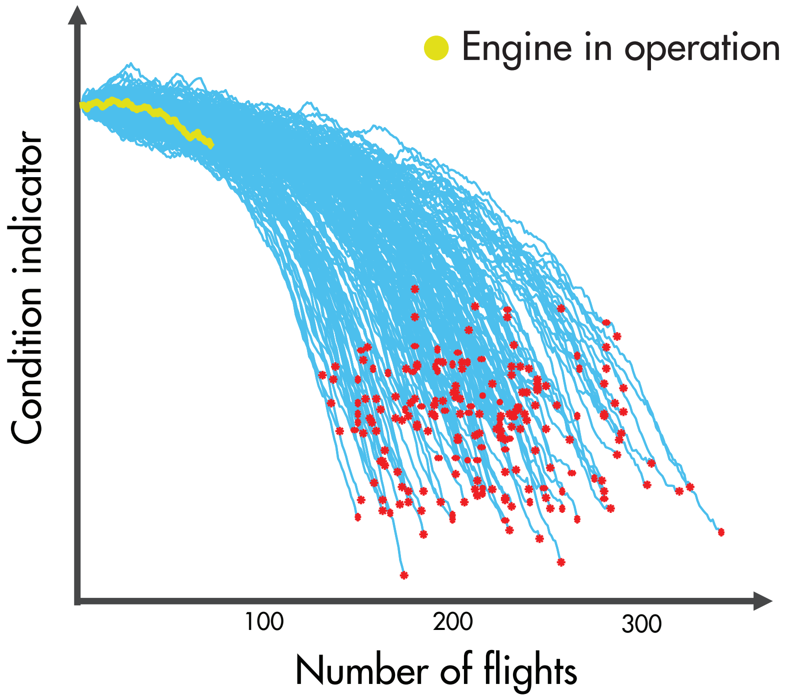
In the preprocessing step, data reduction is performed by selecting only the most trendable sensors (Sensor 2, 11, and 15) and combining them to compute condition indicators.

For more information on how to select and employ condition indicators, see *Predictive Maintenance: Extracting Condition Indicators with MATLAB.*

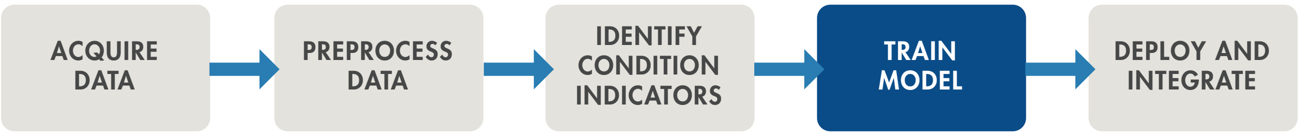
To estimate the remaining useful life for the current engine (shown in yellow), you would use Sensors 2, 11, and 15 to compute condition indicators that represent the degradation profiles of the fleet. In the graph on the right, you can see that the engine is currently at 60 flights, and the red dots mark where similar engines in the fleet have failed.



Combine the most trendable sensors to compute condition indicators



RUL Estimation Workflow Using the Similarity Model *continued*

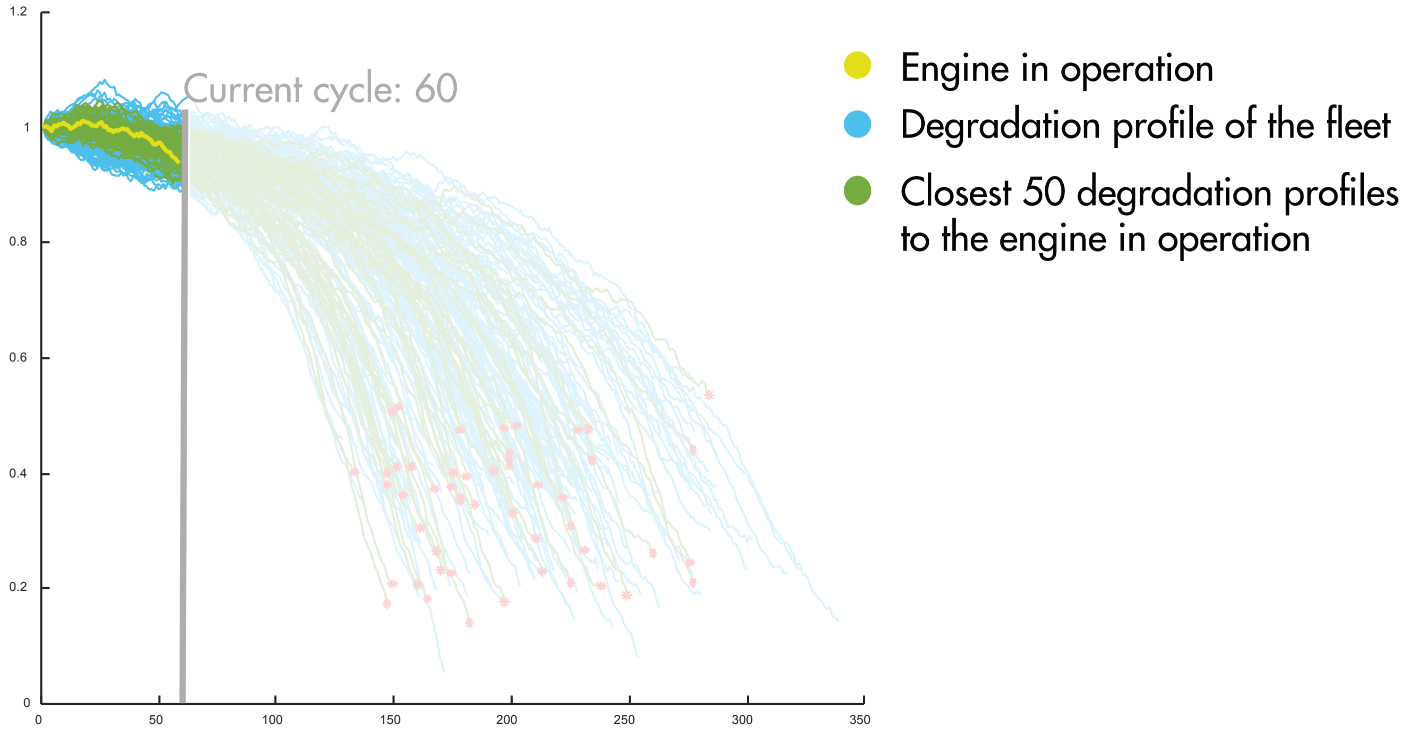


Training a Similarity Model

You'll want to split the data into two groups, using a larger portion of it to train a similarity model and the rest to test the trained model. You use the known RUL to evaluate the trained model's accuracy.

The similarity model works by finding the closest engine profiles to your engine up to the current cycle. You have the failure times of the closest engines from the historical fleet data, which gives you an idea of the expected failure time of your current engine. You can use this data to fit a probability distribution as seen in the bottom plot.

The median of this distribution gives you the remaining useful life estimate of your engine, which you can compare to the actual RUL to measure accuracy.

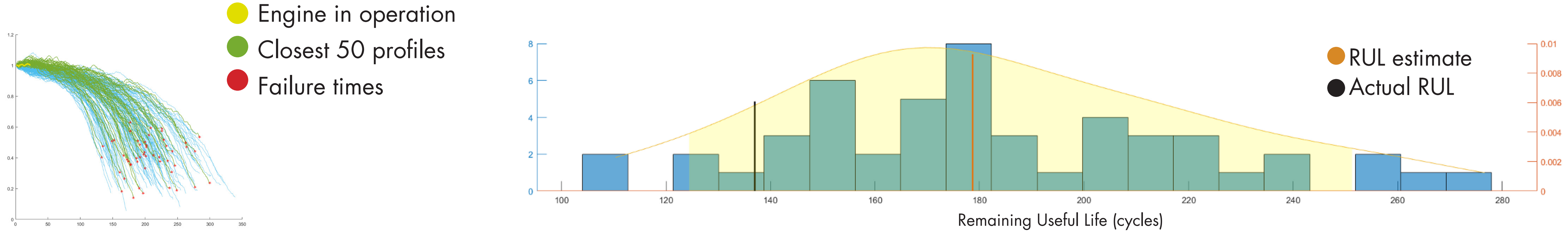


RUL Estimation Workflow Using the Similarity Model *continued*

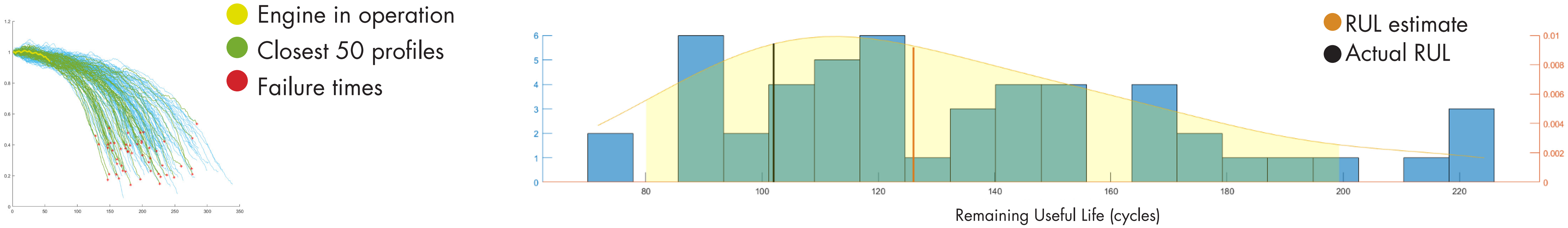
Training a Similarity Model

At each iteration of the RUL computation, the similarity model finds the closest engine paths, which are shown in green, and computes the RUL using a probability distribution plot (shown on the right). The plots below show the engine profiles and their probability distribution for three cycles: 25, 65, and 115.

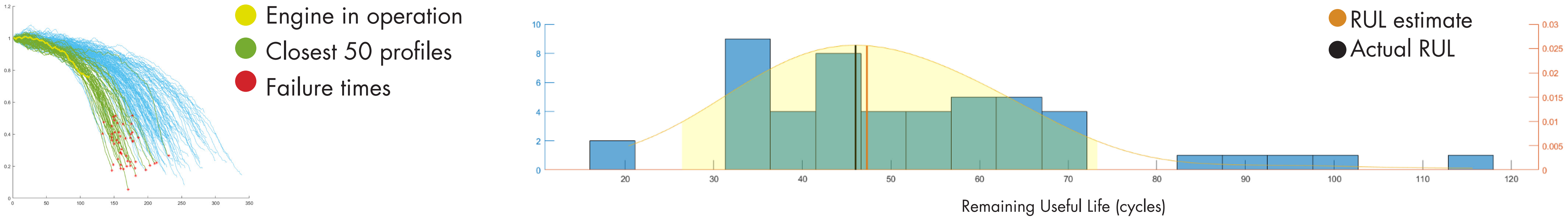
Current Cycle 25



Current Cycle 65



Current Cycle 115



RUL Estimation Workflow Using the Similarity Model *continued*

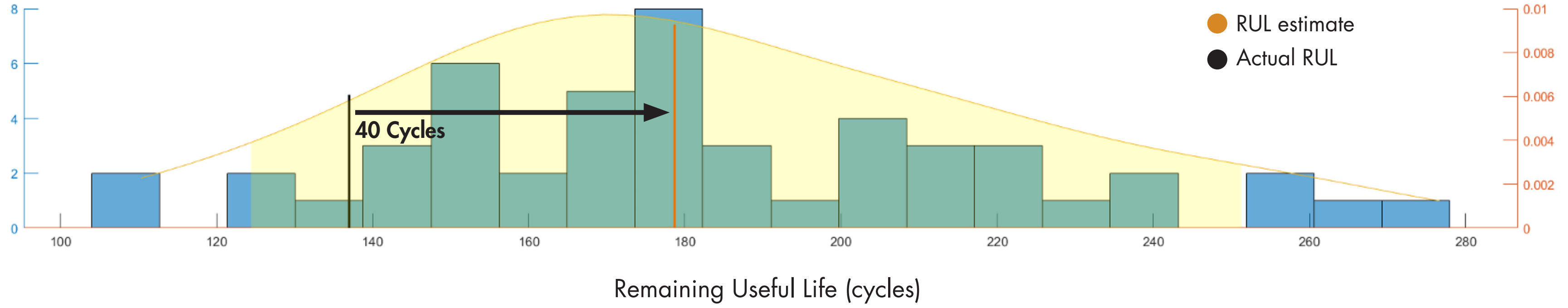
Training a Similarity Model

On the probability distribution plots, the orange lines represent the predicted RULs and the black lines show the actual RUL.

Notice that the predicted RUL gets closer to the actual RUL as time passes and the model has more data.

When the engine is at 25 cycles (or flights), the model doesn't have much data to work with yet, so the prediction is 40 cycles off from the true value with a very wide distribution.

Current Cycle 25

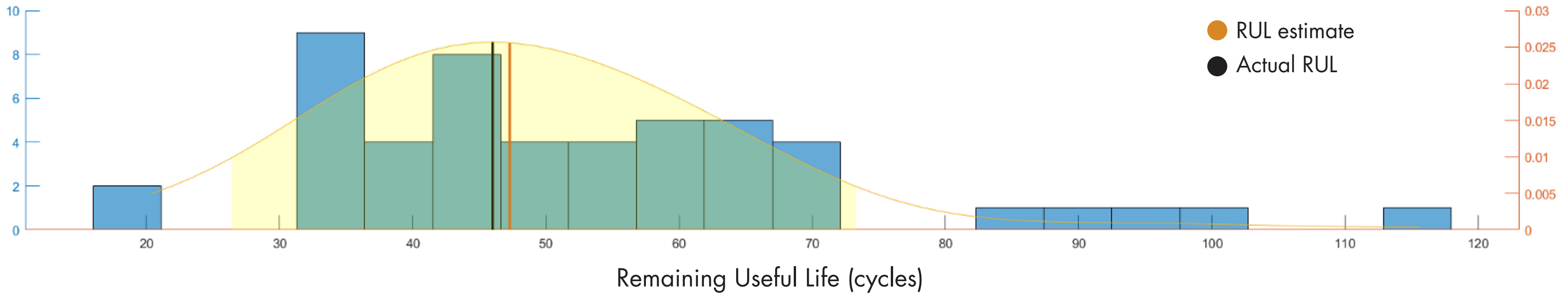


As the model gets new data from the engine, the similarity model trains on a larger set of data, as seen at cycle 65 and 115. As a result, the prediction accuracy improves over time.

With more data the RUL predictions become more accurate and the distribution more concentrated.

Using the RUL estimation from cycle 115, you would have a reasonably accurate expectation of your engine's RUL and be able to schedule maintenance at the best time.

Current Cycle 115



Learn More

Ready for a deeper dive? Explore these resources to learn more about predictive maintenance workflow, examples, and tools.

Watch

What Is Predictive Maintenance Toolbox? (2:06) - Video
Predictive Maintenance Tech Talks - Video Series
Predictive Maintenance in MATLAB and Simulink (35:54) - Video
Feature Extraction Using Diagnostic Feature Designer App (4:45) - Video

Read

Overcoming Four Common Obstacles to Predictive Maintenance - White Paper
MATLAB and Simulink for Predictive Maintenance - Overview
MATLAB Predictive Maintenance Examples - Code Examples