WHITE PAPER

Overcoming Four Common Obstacles to Predictive Maintenance with MATLAB and Simulink

MathWorks®

Predictive maintenance makes a lot of promises, from reducing machine downtime and eliminating unnecessary maintenance to adding revenue streams for equipment vendors with aftermarket services. These benefits are eminently achievable as long as you keep engineering and business challenges from getting in the way.

This paper discusses four common obstacles that stop businesses from successfully implementing predictive maintenance, as identified through more than 100 conversations with engineers and engineering managers. Each of these challenges is solvable; this paper explains how.

1. We do not have enough data to create a predictive maintenance system.

Many predictive maintenance approaches rely on machine learning algorithms, so there must be enough data to create an accurate model. For predictive maintenance, this data usually originates from sensors on machinery. If the sensors are new or the way readings are logged limits the information, you will need to think about the best way to access enough data to build your models.

Take a close look at your list of data sources

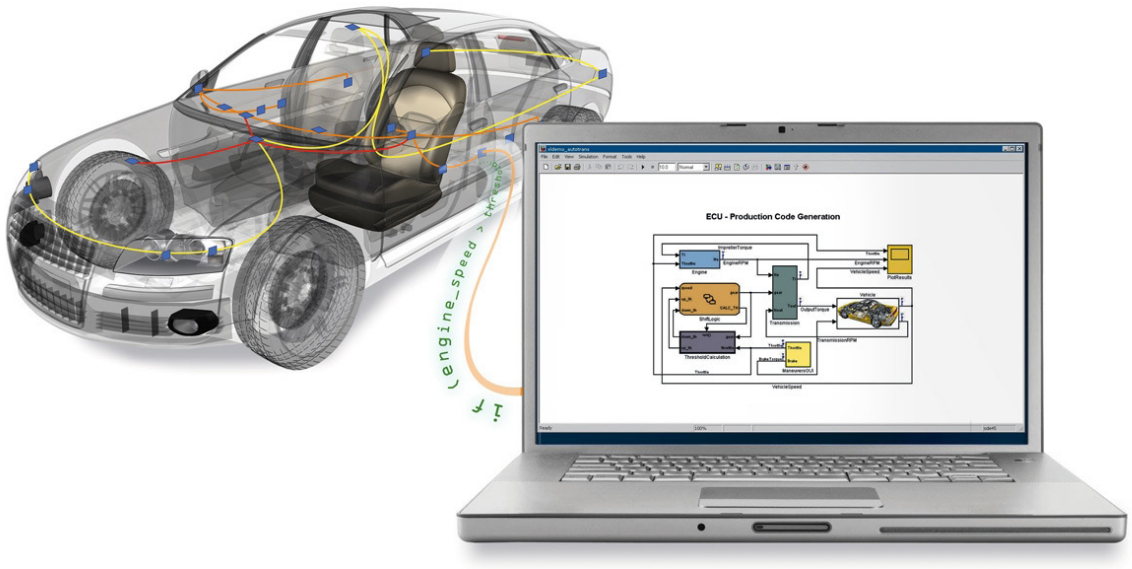
You might find that your department does not collect enough data to power a predictive maintenance system. Consider whether other departments collect data as well. Perhaps the Controls division does not collect enough data, but what if it was combined with data from the Services division? Looking farther afield within your organization could be enough to meet your needs.

Depending on the size of your business and where you are in the supply chain, it is worth looking at your agreements with your suppliers or customers. Cooperating to prolong the health and efficiency of equipment components may put you in a win-win situation that fosters data access between business entities. This will not always be the case, but it is a potential data source that merits consideration.

Change how data is captured

Some systems operate in a "feast or famine" mode where little to no data is collected until a fault occurs. Others only log event codes and time stamps: engineers are notified that an event occurred, but not the sensor values at the time of the failure. Although this data may be useful for diagnostics, it is likely insufficient for developing models that can predict failures.

Consider changing the data logging options to record more data, perhaps on a test fleet if production data is not available. Depending on the load on existing embedded devices, it may be possible to reconfigure them to collect and transmit sensor data, or external data loggers may be necessary to get started.



Configuring data logging to collect and transmit sensor data.

Use simulation tools to synthesize data

Generate test data using simulation tools and combine that data with what sensor data is available to build and validate predictive maintenance algorithms. This is done by creating models that cover the mechanical, electrical, or other physical system to be monitored. Synthesize sample data (by modeling output readings) and validate this against measured data to ensure the model is well-calibrated. This can be done at the component level first, then later at the system level for complex systems.

Mazda Simulates Data for Engine Development

Mazda needed to define test plans, develop statistical models, and generate optimal calibrations for their SKYACTIV-D engine. They developed statistical models for the SKYACTIV-G and performed hardware-in-the-loop (HIL) simulation of engine control logic.

> *"With traditional methods, getting data when calibrating a new engine required a large amount of testing…We reused the existing data and simulated the responses, which enabled us to minimize both the workload to obtain test data and test cell usage."*
>
> **— Shingo Harada, Mazda**

» ***Read user story***

Considerations

When considering data for a predictive maintenance system, begin to analyze data early to understand which features are important and which may be redundant. Depending on where you store your data, it can be expensive to keep an excessive amount of data that is not going to be used. Once you understand which features of the data are the most important, you can make informed decisions about which data needs to be kept and which does not. One benefit of a tool like *MATLAB®* is that it is decoupled from the storage system; if you move from local to cloud storage, you can still run your analysis with minimal changes.

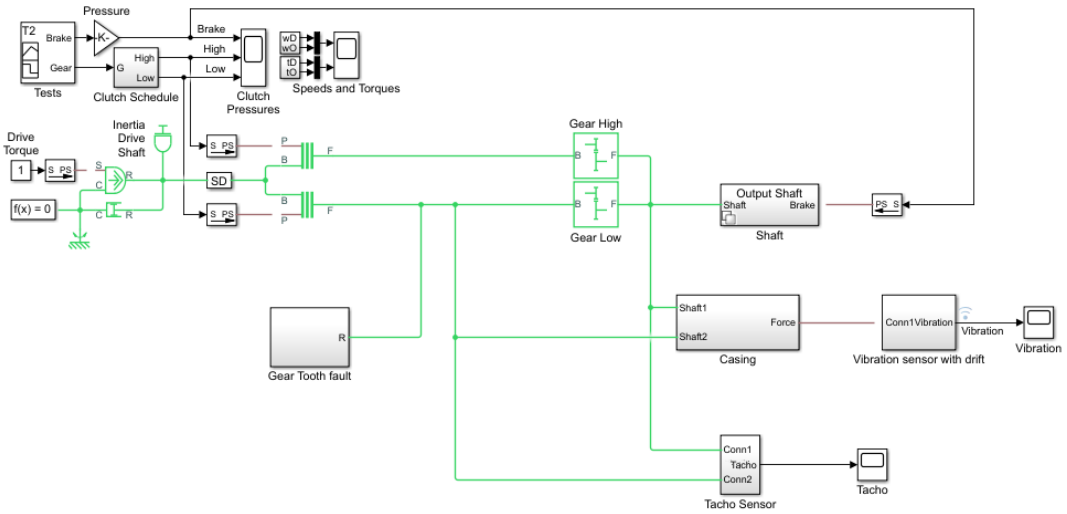
2. We lack the failure data needed for accurate results.

Failure data is a crucial part of teaching algorithms to recognize the warning signs to trigger just-in-time maintenance. Failure data may not exist if maintenance is performed so often that no failures have occurred, or the system is safety critical and cannot be left to fail. To stop this from becoming a fatal deficiency, you and your team can simulate failure data and learn how to recognize warning signs from available operations data.

Generate sample failure data

An engineer with deep system knowledge of how the physical components work will be able to generate sample failure data with the right tools. Using a simulation product like *Simulink®*, an engineer can build or use a physical model of the machine as described in the first challenge. Tools such as failure mode effects analysis (FMEA) provide useful starting points for determining which failures to simulate. An engineer with sufficient domain knowledge can incorporate these behaviors into the model in a variety of scenarios, which simulate failures by adjusting temperatures, flow rates, or vibrations or adding a sudden fault. These scenarios can then be simulated, and the resulting failure data is labeled and stored for further analysis.

Products like *Predictive Maintenance Toolbox™* simplify tasks like *failure data generation* and provide data ensembles for managing and organizing multiple datasets.



Using Simulink to generate fault data.

Airbus Models Multiple Component Failures for the A380 Commercial Aircraft

Airbus needed to safely handle failures in the complex fuel management system for the A380. The team simulated failures to refine the model. After successful flight tests, they evaluated differences between the measured data and predicted results to further tune the models.

"Model-Based Design gave us advanced visibility into the functional design of the system. We also completed requirements validation earlier than was previously possible and simulated multiple simultaneous component failures, so we know what will happen and have confidence that the control logic will manage it."

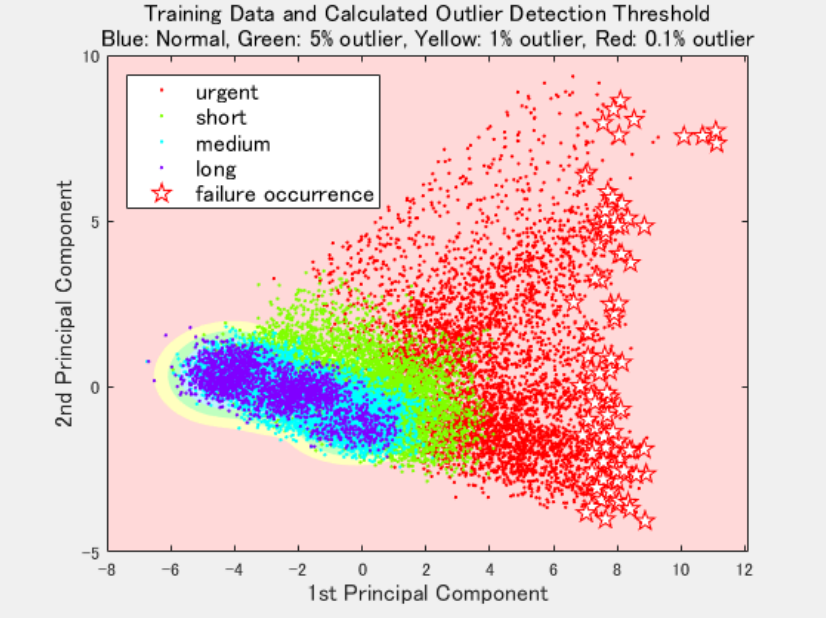
— Chris Slack, Airbus

» *Read user story*

Understand the data available

Failure data might not be present, but operations data might show trends about how a machine degrades over time.

Looking at the raw sensor data from a component, system, or machine with dozens or hundreds of sensors can be intimidating. Statistical techniques such as principal component analysis (PCA) can help reduce the dimensionality of such datasets and provide valuable insight into how equipment operates over time. PCA is one of many unsupervised learning techniques. Unsupervised learning is a branch of machine learning that attempts to find patterns and trends in unlabeled data. Depending on what sensors are available, certain types of failures may require looking at several sensors simultaneously to identify undesirable behavior. Unsupervised learning techniques transform raw sensor data into a lower-dimensional representation, which can be visualized and analyzed much more easily than the high-dimensional raw data.



Using principal component analysis to visualize how equipment trends prior to failure.

Considerations

Try to keep the number of variables down to the minimum needed for an accurate model. It can be tempting to include every measured component to make sure nothing is missed, but this will result in a black-box model that is overwrought with complexity. Techniques like PCA avoid this situation and are a quantitatively rigorous method to achieve this simplification.

3. We understand the failures, but we cannot predict them.

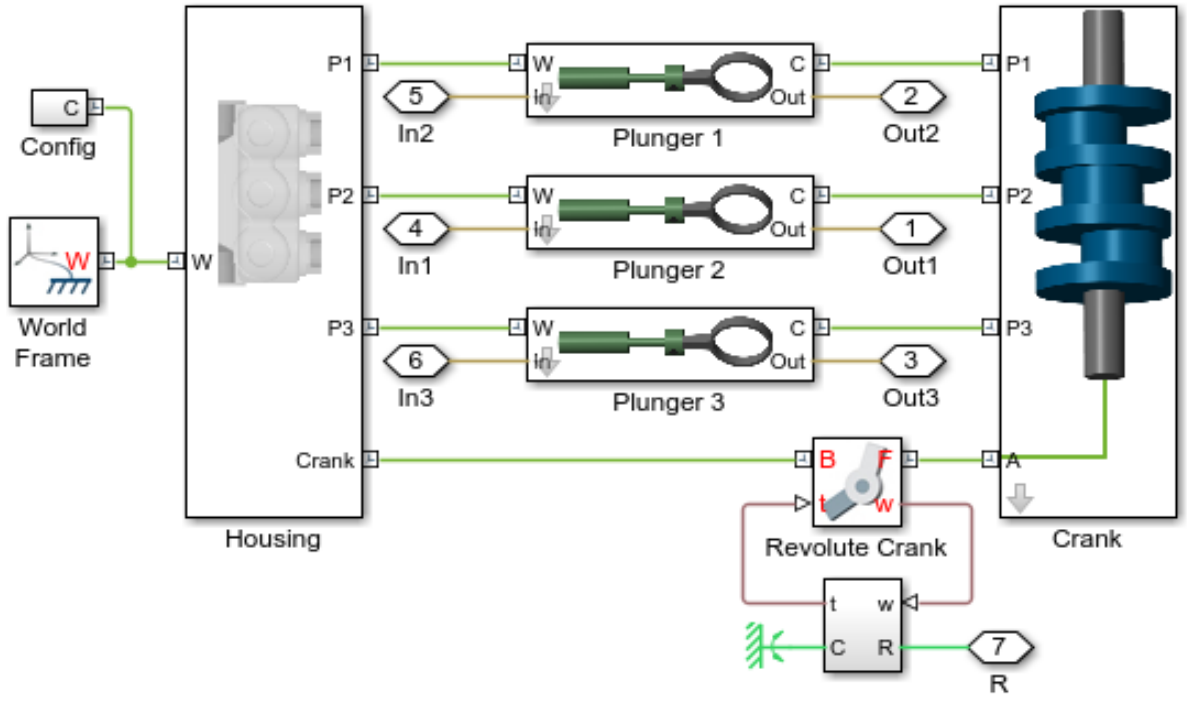
Understanding the cause of a failure is important for your business, but there is a significant difference between identifying what went wrong and knowing how to predict it. Root cause analysis is an integral part of domain knowledge that, paired with predictive maintenance algorithms, creates an effective predictive maintenance program. If the algorithm part of the equation is a new and intimidating undertaking, you can take these steps to reduce the learning curve.

Define goals

How do you know if a predictive maintenance algorithm is better than the previous way of doing things? It is important to define upfront what your goals are (e.g., earlier identification of failures, longer cycles, decreased downtime). You should then think about how the predictive maintenance algorithm will affect these goals. Building a framework that can test an algorithm and estimate its performance relative to your goals will enable faster design iterations. It will no longer be up for debate whether a new algorithm is better than the previous state, but rather it will be clear if a new algorithm is better based on the agreed-upon goals.

Start small

If you and your team already know the causes behind failures, then the domain knowledge is there. Choose a project using a deeply understood system to practice on. Make sure you understand the features and factors that affect the performance of the system, and build a predictive maintenance algorithm. As the simplest starting point, it is worthwhile to consider if thresholding a feature is a significant maintenance indicator (typically done via control charts). The domain knowledge your team has will help with principal component identification as well as threshold values, such as a safety value that should never be exceeded. Beyond that, you may try simple models such as linear or logistic regression, which are quick to fit and easy to interpret. Once you and your team are comfortable with building the algorithms for a simple problem, you can apply that knowledge to more complex systems.



Modeling three types of faults: cylinder leaks, blocked inlet, and increased bearing friction.

Gain confidence

When predictive maintenance algorithms begin to show promising results, use current and historical data to test and validate models before moving to production. Use the domain knowledge within your team to tune models to predict different outcomes based on the cost/severity of those outcomes. To further validate models, add generated failure data similar to known historical conditions and test the system. This validation step will build confidence that the process is working, whether by highlighting where the simulation doesn't match reality and needs more work, or by endorsing the model's accuracy.

Considerations

As with any new task, it is important not to try to do everything at once, only to get frustrated when the project feels too complex. Define clear goals, start small, validate against data, and iterate until you are confident with the results. Repeat this process to build up to more complex systems.

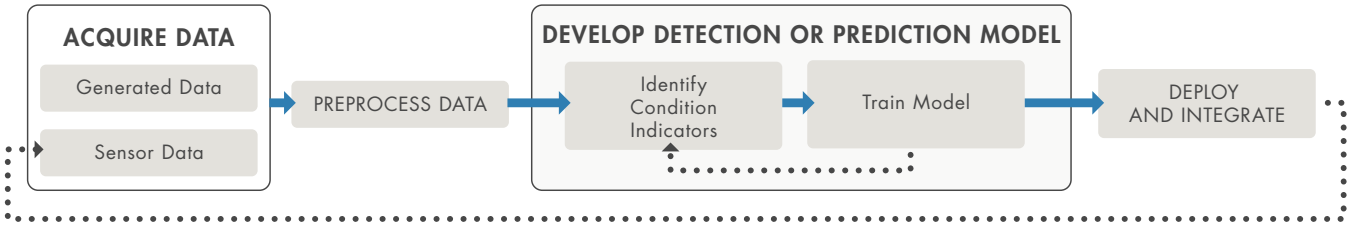
4. We do not know how to do predictive maintenance.

Every new technology requires investment that must be justified. Ideally, the time required to realize the value of the investment should be as short as possible. Quantifying how long until you see a return on investment is difficult when there are uncertainties about how quickly you and your team can become adept in these new skills. If machine learning has only recently been introduced, it is only natural to see what might be considered an advanced application of it as a risk. However, you can take concrete steps to minimize that risk and get up and running with a working predictive maintenance model as quickly as possible.

Work with the tools your engineers already know

Instead of trying to introduce a new technology and technique, take advantage of new capabilities in the software already in place and focus on the new techniques. Some tools that engineers already use, such as MATLAB, have specific predictive maintenance capabilities, enabling engineers to continue working in an environment they know. These tools also provide reference examples and algorithms to help people new to predictive maintenance get up and running quickly as well as technical support, training, and consulting teams. The additional guidance can get the basics in place so you and your team can be confident that you approach problems in the best way.

Learn the predictive maintenance workflow



The basics of the predictive maintenance workflow.

The first step to getting started producing a working model is to understand the workflow and recognize what might slow progress. To build and deploy predictive maintenance algorithms, there are five stages:

1. Access sensor data

Data can be gathered from multiple sources, such as databases, spreadsheets, or web archives. Make sure the data is in the right format including date and time stamps. Large data sets may not fit into memory, and will require out-of-memory processing techniques or a cluster. Pain points are often around how to organize the data for analysis. If you don't have enough data, you can generate data from a physical model of the machine to supplement normal usage, varying parameter values, different system dynamics, or signal faults.

2. Preprocess data

Data in the real world is rarely perfect; it has outliers and noise that need to be removed to get a realistic picture of normal behavior. If the data has come from different sources, it will also need to be combined. If you remove anomalies, think about whether you want to replace them with approximate values or work with a smaller data set. Potential pain points include adjusting noise filtering or outlier settings or comparing the effect of different filtering on overall algorithm performance.

3. Extract features

Instead of feeding sensor data directly into machine learning models, it is common to extract features from the sensor data. These features capture higher-level information in the sensor data, for example moving averages or frequency content. Few engineers have a strong background in statistics, signal processing, and system modeling, so the use of familiar tools to perform feature extraction techniques simplifies this step. An iterative approach—in which features are added, new models are trained, and their performance is compared—can work well here to determine the effectiveness of different features on the results.

Baker Hughes Extracts Features to Train Models for Oil and Gas Extraction Equipment

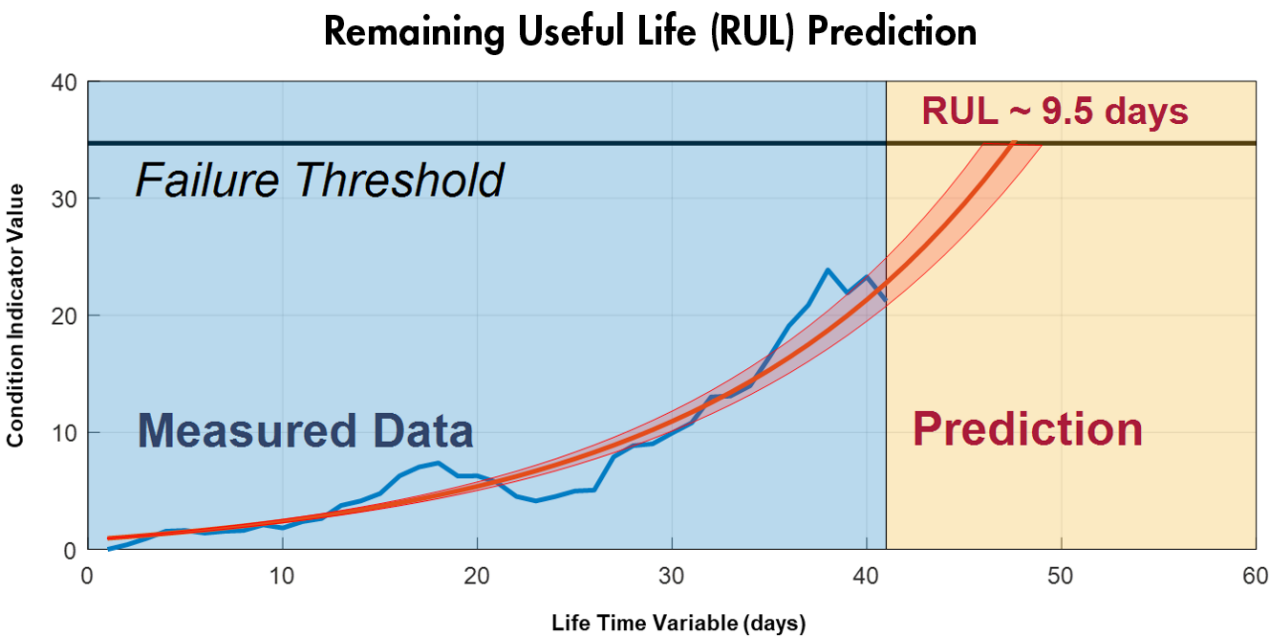
Working in MATLAB, the Baker Hughes team analyzed the data imported from gas and oil extraction equipment to determine which signals in the data had the strongest influence on equipment wear and tear. This step included performing Fourier transforms and spectral analysis as well as filtering out large movements of the truck, pump, and fluid to better detect the smaller vibrations of the valves and valve seats.

> *"MATLAB gave us the ability to convert previously unreadable data into a usable format; automate filtering, spectral analysis, and transform steps for multiple trucks and regions; and ultimately, apply machine learning techniques in real time to predict the ideal time to perform maintenance."*
>
> **— Gulshan Singh, Baker Hughes**

» ***Read user story***

4. Train the model

In this step, you classify data as healthy/faulty, set thresholds for healthy/warning/failure states, and estimate remaining useful life (RUL) for components. You'll need to create a comprehensive list of failure scenarios to predict, choose classification methods, and simulate models. Apps provide graphical interfaces for applying machine learning that make it easy to get started and compare the results of training many different types of models.



Training predictive models that can estimate remaining useful life and provide confidence intervals associated with the prediction.

5. Deploy the model

Generate code and deploy models as an application on hardware. Models may be deployed to embedded devices by converting them to a low-level language such as C, or they may be integrated with other applications in an IT environment. The pain here is often around lack of familiarity with code generation and IT integration. There are tools that can automatically package models to run in a production environment like ***MATLAB Compiler***™ and ***MATLAB Production Server***™. Consulting can be particularly useful when trying to integrate these applications into IT systems.

Mondi Develops a Predictive Maintenance System

Mondi Gronau's plastic production plant runs 24/7. With the help of MathWorks Consulting, they created a health monitoring and predictive maintenance application that enables plant workers to take corrective action and prevent serious problems. They completed it in six months and are saving an estimated 200,000 euros per year.

"MathWorks Consulting's support is among the best I've seen; the consultants are fast and exceptionally knowledgeable. We've already seen a positive return on investment from cost savings, and now we have more budget and time to complete more machine learning projects that will provide similar benefits."

— Dr. Michael Kohlert, Mondi

» ***Read user story***

Considerations

Some businesses need daily reports on their machines, while others need real-time processing. Think about what level of supervision is necessary to your business. Also keep in mind the data types you are collecting—is it signal, image, or text data and could it be helpful in predicting failures? You will need the computing power to accommodate large volumes of data. Finally, consider how you need the results presented, how much insight into how warnings are determined is necessary, and who needs to receive these notifications.

Conclusion

Predictive maintenance is an achievable goal with the right tools, guidance, and motivation. Find the features, models, and methods that work for your business and iterate until you get it right—and remember you do not have to do it alone.

Learn More

- ***Predictive Maintenance with MATLAB: Avoid Costly Equipment Failures by Using Sensor Data Analytics*** - Ebook
- ***Machine Learning with MATLAB*** - Ebook
- ***MathWorks Consulting for Predictive Maintenance*** – Service Overview