Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Simulación del deterioro de las máquinas a través del mini-término en líneas de producción

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: David Sabater González

Tutor: J. Alberto Conejero

2020/2021

Resumen

Este proyecto mostrará el desarrollo de un simulador que permita generar datos, como los que se obtendrían al monitorizar máquinas de una cadena de montaje. Estos datos permitirán entrenar sistemas de aprendizaje y que se detecten patologías en máquinas a partir de los mini-términos.

Este simulador dispondrá de un entorno gráfico que permita configurar los parámetros de cada modelo y mostrará los resultados de la simulación de forma gráfica tras la ejecución, permitiendo que se decida si los datos mostrados deben, o no, guardarse.

Palabras clave: simulador, mini-término, cadena de montaje, mantenimiento, patologías.

Abstract

This project will show the development of a simulator that allows the generation of data, such as those that would be obtained when monitoring machines in an assembly line. This data will allow to train learning systems and to detect pathologies in machines using mini-term.

This simulator will have a graphical environment which will allow to configurate each model's parameters and will show simulation results graphically after execution, allowing the user to decide if shown data should, or shouldn't be, saved.

Keywords :. Simulator, mini-term, assembly line, maintenance, pathologies.

Tabla de contenidos

Índice de imágenes y tablas

1 Introducción

Una línea de producción se puede definir como el conjunto de operaciones secuenciales en las que se organiza un proceso para la fabricación de un producto. A través de fases y operaciones se realiza la transformación de materia prima en el producto. En cada fase hay unos trabajadores y/o maquinaria asignados a un proceso. Estas fases realizan sus tareas por separado en el montaje final y, eventualmente, a la fabricación del producto final.

Las líneas de producción deben estar equilibradas. De tal forma, que se organicen para que la asignación de puestos de trabajo y máquinas sea en función del tiempo asignado a cada operación. El objetivo es que en cada operación se ocupe el máximo tiempo de trabajo del operario y no se produzcan cuellos de botella.

En cada operación se organizan operarios y máquinas especializados para esta. Consiguiendo un menor tiempo de producción y una mayor productividad. Cada operación de la cadena realiza turnos en sus operarios y, de esta forma, la línea de producción puede estar activa continuamente. Aunque los empleados descansen, las máquinas están continuamente trabajando y, eventualmente, se desgastan. Cuando una máquina falla, necesita un mantenimiento y se está perdiendo capacidad de producción hasta que este se realiza.

Desde que el 7 de octubre de 1913, en la planta de Ford en Piquette Avenue (Detroit), se empezara a producir el Ford Model T en masa, con una cadena de montaje inspirada en las cadenas que Henry Ford había visto en el matadero de su ciudad, según cuenta la leyenda. A partir de este momento, comenzó un cambio, el mundo entró en el fordismo.

Sin embargo, el montaje en cadena había sido establecido y patentado antes, en 1901, por Ransom Eli Olds y, por otro lado, Henry Leland destacaba con su estandarización de componentes para los Cadillac. Estas influencias sentaron unas bases imprescindibles para la eclosión de la industria automovilística.

Lo que hizo destacar al equipo de Ford sobre sus competidores fue el empleo de mano de obra no cualificada, granjeros que acudían a la ciudad para buscar fortuna y que se convertirían en obreros de automóviles. La clave del sistema está en subdividir las tareas hasta la mínima expresión y llevar los componentes hasta los trabajadores, en vez de esperar a que cada trabajador se desplazara hasta el vehículo que fabrica.

Con el paso del tiempo, esta técnica se fue modernizando. Se añadieron máquinas para realizar operaciones y los operarios pasaron a centrar más su trabajo sobre estas máquinas, ya que necesitan de revisión y mantenimiento. Para seguir con la filosofía de optimizar la producción, y hacer el trabajo lo más sencillo posible, se busca mejorar la claridad de qué operaciones de mantenimiento son necesarias y cuando.

Las operaciones de mantenimiento tienen una influencia directa en la capacidad de producción. Es importante desarrollar una buena planificación y priorización de tareas de mantenimiento, sobre todo teniendo en cuenta que los recursos de mantenimiento son limitados.

Las operaciones de mantenimiento se pueden clasificar en dos grandes clases: mantenimiento correctivo (CM) y mantenimiento preventivo (PM). El CM se realiza cuando una máquina falla. Normalmente implica el cambio o la reparación de un componente. Por otro lado, el PM se realiza antes de que la máquina falle. Con esto se espera conseguir un sistema de producción continua. Normalmente, se puede predecir la necesidad de este mantenimiento a través de parámetros como la vibración, la temperatura, el nivel de ruido, etc.

Sin embargo, monitorizar constantemente estos parámetros puede suponer un incremento del coste del sistema si se necesitan dispositivos de monitorización caros.

Sería posible conseguir un resultado similar al análisis proporcionado por estos dispositivos, pero ahorrando el coste que suponen, a partir de la utilización de mini-término.

1.1 Motivación

Con este proyecto se pretende ayudar a un modelo de investigación que busca encontrar un punto técnicamente factible, con el que lograr la mayor productividad y cumplir los objetivos de la compañía para una mayor rentabilidad. Y, con ello, dar apoyo a un proyecto académico innovador y en desarrollo. Una iniciativa coliderada por el profesor Nicolás Montes, investigador principal del Grupo de Automatización Industrial y Robótica de la CEU UCH y Eduardo García Magraner, gerente del área de Ingeniería de Carrocerías y Prensas de Ford Valencia.

Este proyecto de investigación está respaldado por el *University Research Program* (URP) de *Ford Motor Company*. Esta iniciativa de Ford tiene como objetivo desarrollar nuevas tecnologías y generar conocimiento valioso para la compañía a nivel global.

Para realizar este estudio se han utilizado las técnicas de mini-término, la herramienta para monitorizar los cambios sin hardware adicional ni dispositivos de monitorización caros (a través de la PLC y sensores instalados para la automatización de la producción).

En última instancia, se pretende conseguir la conjugación correcta de tres factores, producción, calidad y mantenimiento. La clave de la competitividad de la compañía Ford.

Tras la realización de este proyecto, se espera poder preparar la maquinaria de la línea de producción para anticiparse a ciertos problemas que puedan surgir en esta. Problemas tales como desgaste de componentes, trozos de metal acumulados durante la labor, falta de lubricante, etc.

Esta información permitirá una mejora del rendimiento de líneas de producción automatizadas ya existentes y en funcionamiento. Al saber con antelación cuando ocurrirán estos problemas, y qué problema en qué momento, se puede estar preparado para solucionarlo al momento. Esto reduce el tiempo gastado en mantenimiento, que es un cuello de botella de la producción.

Un cuello de botella se define como la estación con mayor tiempo de ciclo. Es un impacto clave en el rendimiento de la línea de producción. Este tiempo extra provoca un bloqueo a estaciones anteriores y un tiempo de espera a estaciones siguientes. Teniendo esto en cuenta, al referirse al mantenimiento en estaciones de la línea de producción, esta pausa en el ritmo de producción puede cambiar de estación. Se convierte en un ente dinámico. Por lo tanto, anticiparse a esta situación puede suponer que el tiempo de bloqueo y el tiempo de espera de otras estaciones sea mínimo.

Con esta mejora en la eficiencia de la cadena se espera aumentar la productividad general y la optimización en el uso de los recursos utilizados en las labores de mantenimiento.

1.2 Objetivos

En este proyecto se deben cumplir los siguientes objetivos:

Diseñar una herramienta que realice una simulación de máquinas de una cadena de montaje y genere los datos que se obtendrían de una detección realizada en un entorno real.

Los datos generados deberán poder guardarse como un Excel para que más tarde puedan ser utilizados para entrenar un sistema capaz de monitorizar en tiempo real el deterioro de las máquinas y predecir su avería.

El desarrollo deberá estar basado en el lenguaje de programación R.

Es necesario que, al finalizar, este proyecto disponga de una interfaz gráfica. con la que configurar la simulación a ejecutar. Además, la interfaz, deberá mostrar los resultados tras la ejecución a través de dos gráficas para proceder a guardarlos si se desea.

1.3 Estructura del documento

Este documento se ha organizado en ocho capítulos. La estructura seguida se explicará en este apartado.

El primer capítulo es la introducción, que aportará información sobre la motivación y la estructura del documento.

El segundo capítulo es un estudio de la situación actual en el entorno del mantenimiento, sobre todo el mantenimiento preventivo. A continuación, se hace una visión a otras técnicas similares a la relacionada con el proyecto y se dará un vistazo a cada una. Por último, en este capítulo, se presenta la propuesta de este proyecto.

El tercer capítulo es un análisis de los requisitos del proyecto. Se realiza un acercamiento a los propósitos y el primer objetivo a desarrollar. También se analizan los tipos de simulación y sus parámetros. Y, para finalizar el capítulo, se compara las soluciones tomadas para el desarrollo con soluciones alternativas que podrían realizar un resultado similar.

El cuarto capítulo representa todo el diseño realizado para desarrollar el sistema. Se representan dos aspectos la parte gráfica y la lógica.

El quinto capítulo está dedicado al desarrollo del proyecto. También se habla sobre el software utilizado.

El sexto capítulo es la implantación, se comentan los requisitos del sistema que va a ejecutar la simulación desarrollada.

El séptimo capítulo son pruebas realizadas.

El octavo capítulo es un repaso a cómo ha finalizado el proyecto conforme a los objetivos formulados inicialmente.

Por último, el noveno capítulo es una muestra de referencias bibliográficas.

2 Estado del arte

2.1 Crítica al estado del arte

El objetivo de este apartado es mostrar, y evaluar, las opciones a la hora de actuar y gestionar el mantenimiento de una cadena de montaje evaluando ventajas y desventajas de cada una de estas.

Como se mencionó anteriormente, al hablar de mantenimiento, puede clasificarse en dos grupos principales: Mantenimiento Correctivo (CM) y Mantenimiento Preventivo/Predictivo (PM).

2.1.1 Mantenimiento Correctivo (CM)

El término hace referencia al primer concepto de mantenimiento y el único hasta la Primera Guerra Mundial, una época en la que se contaba con máquinas, equipamientos e instalaciones más simples.

Se trata, por lo tanto, del tipo de mantenimiento más básico y menos eficiente para la producción. Se realiza al observar un defecto o avería para corregirlo o repararlo, por lo tanto, indica una situación inesperada y la respuesta es más lenta e ineficaz de lo que podría conseguir con el PM. Además, presenta costos por reparación y repuestos no presupuestados.

2.1.2 Mantenimiento Preventivo/Predictivo (PM)

Tras la Primera Guerra Mundial se empieza a estudiar que el mantenimiento no se limitara a corregir averías, sino que se adelantara a estas garantizando un correcto funcionamiento y evitando retrasos.

Se trata, por lo tanto, del tipo de mantenimiento que se lleva a cabo antes de que el equipo falle, con el objetivo de promover la producción continua del sistema y/o minimizar la pérdida de rendimiento.

A su vez, el mantenimiento preventivo/predictivo se puede dividir en dos grandes tipos de estrategias, las basadas en tiempo (Time-based Maintenance, TBM) o las basadas en el estado de la máquina (Condition-based Maintenance, CBM).

a. Time based Maintenance, TBM

La estrategia de mantenimiento basada en el tiempo (TBM) consiste en realizar un mantenimiento preventivo periódicamente., lubricando, calibrando y realizando inspecciones periódicas.

Al aplicar esta estrategia se suele actuar conforme a recomendaciones del fabricante, histórico de fallos, experiencia de los operarios y/o del staff de mantenimiento.

b. Condition-based Maintenance, CBM

La estrategia de mantenimiento basada en "condiciones", del sistema y sus componentes, (CBM) consiste en hacer un diagnóstico en tiempo real para determinar si sería necesario la aplicación de alguna acción previa a un fallo próximo.

El objetivo es evitar tareas innecesarias de mantenimiento y solo realizarlas cuando exista una evidencia de funcionamiento anormal. Es una estrategia proactiva que requiere del desarrollo de un modelo predictivo.

La motivación del CBM es que el 99% de los fallos de los equipos vienen precedidos por ciertos signos, condiciones o indicaciones de que el fallo está a punto de ocurrir. Para detectar estos signos, se utilizan medidas de sensores tomadas de forma periódica e incluso de forma continua. En general, el CBM cumple dos propósitos. Recoger los datos del estado de la máquina y segundo incrementar el conocimiento de las causas de los fallos, de los efectos y de los patrones de deterioro de los equipos.

Además, a través de esta estrategia, se puede asegurar una alta calidad del producto final, sobre todo si se seleccionan bien los umbrales de medida que se están tomando de la máquina.

El CBM se puede llevar a cabo con las máquinas activas o paradas. En el caso de realizarlo con la máquina detenida, es común buscar grietas, cambio de color, etc. También, se puede llevar a cabo de forma periódica o continua. La forma más común de usarlo es de forma periódica, por ejemplo, cada hora o cada cambio de turno, aunque la forma ideal sería utilizarlo de forma continua y automática. Sin embargo, no se suele realizar de forma continua porque el coste para los sensores y dispositivos necesarios es muy caro. Los sensores más utilizados en la CMB son:

- **Vibración:** El uso de sensores de vibración es una de las técnicas más usadas para el CBM, especialmente para máquinas con elementos rotatorios. El análisis se realiza in-situ, requiere que el sensor esté en contacto con la máquina y es un test no destructivo. Con estas mediciones se pueden detectar fallos tales como desequilibrio, desalineación, holguras, roces, etc.

- **Ruido:** El análisis de los niveles de ruido es otra de las técnicas más utilizadas en el CBM. Tiene una fuerte relación con la vibración y, por tanto, también se utiliza para máquinas con elementos rotativos. Al contrario que los sensores de vibración, estos no requieren estar en contacto con la máquina o elemento a sensorizar, la monitorización del ruido simplemente está "escuchando" el equipo.

- **Análisis del aceite o lubricante:** Esta técnica es fundamental para determinar el deterioro del lubricante, la entrada de contaminantes y la presencia de partículas de desgaste. Además, proporciona una medida indirecta del nivel de deterioro de los componentes a los que lubrica. Actualmente, existen equipos de taller que permiten realizar un análisis rápido de aceites en la planta industrial para reducir el coste de análisis por muestra y conseguir resultados inmediatos.

- **Medidas eléctricas:** Esta técnica busca cambios en las propiedades de los equipamientos, tales como resistencia, conductividad y aislamiento. Esta técnica suele emplearse para detectar deterioro de aislamiento en motores.

- **Temperatura:** También llamada termografía, esta técnica se utiliza fundamentalmente para detectar fallos en componentes eléctricos. La medida de temperatura sin contacto es una técnica fundamental que ha experimentado grandes cambios en los equipos e instrumentación disponibles, y que está en continua evolución. En esta, se usan cámaras termográficas y gracias a su reducción en los precios están al alcance de cualquier departamento de mantenimiento.

- **Presión, caudal, consumo eléctrico:** Son técnicas usadas, aunque menos que las anteriores.

Bajo el concepto CBM se pueden tomar dos decisiones: Diagnóstico y predicción. Diagnóstico es el proceso de encontrar la fuente del fallo, mientras que predicción es el proceso de estimar cuando se producirá el fallo. Con el diagnostico se alerta al personal de mantenimiento de que el equipo está funcionando en condiciones anormales, aunque esto no quiere decir que el equipo haya fallado. Sin embargo, advierte que fallará después de un cierto tiempo. Este tiempo es estimado a partir de la predicción. Desde el punto de vista del mantenimiento, la predicción es mucho más relevante que el diagnostico porque puede predecir fallos inesperados.

Un concepto esencial en la predicción es el *change point*, es un indicativo de que algo anómalo está pasando y que el fin de la vida útil de algún componente está cerca. Esto se detecta como un cambio abrupto de la medida que se esté realizando de la máquina y siempre está relacionado con algún cambio físico del componente. En el caso del aceite o lubricante, cuando llega al final de su vida útil, su viscosidad cambia de forma abrupta y se detecta un cambio brusco en el rendimiento. Cuando un componente o pieza es sometida a una carga constante la elongación que sufre al final se convierte en un alargamiento acelerado, el proceso se conoce como "*the creep curve*". También ocurre algo parecido con el coeficiente de elasticidad, cuando una pieza es sometida a flexiones continuas, llega un momento al final de su vida útil en que no recupera su posición inicial.

El reto principal de las políticas CBM es el remanente de la estimación de la vida útil de los equipos, *Residual Useful Life (RUL)*. Los pronósticos sobre la vida útil son muy interesantes para mejoras en los aspectos ambientales, económicos y operativos. En el marco de la gestión de pronóstico y la salud, *Prognostics and Health Management* (PHM) existen muchas técnicas de pronóstico y básicamente se clasifican en cuatro categorías principales: enfoques basados en el modelo físico (*Physical model based methology*), enfoques basados en la experiencia (*Knowledge-based methodology*), los enfoques basados en datos (*Data driven methodology*) y los modelos combinados.

Los métodos basados en el modelo físico utilizan el modelo matemático relacionado con la salud de la máquina directa o indirectamente. Son diseñados por expertos y son específicos para la máquina a monitorizar. Las ventajas de esta técnica son que es posible establecer relación entre los parámetros monitorizados y las patologías a detectar y que, con una buena comprensión del proceso de degradación, es posible adaptar el modelo para mejorar su precisión. Por otro lado, su principal limitante es que están diseñados específicamente para una máquina concreta.

Los métodos basados en experiencia son la principal alternativa a los métodos basados en el modelo físico. Los sistemas expertos tratan de "digitalizar" las tareas que usualmente realizan los especialistas y su forma de tomar decisiones, combinando el potencial de las computadoras con las leyes del razonamiento. Esta metodología busca soluciones utilizando lógica difusa, redes neuronales, etc. Por otro lado, no puede cubrir eventos que no han sido explícitamente especificados en las reglas. En adición a esto, un número elevado de reglas produce una "explosión combinacional", provocando problemas computacionales.

Los métodos basados en datos utilizan la estadística y las técnicas de aprendizaje, implementadas en gran parte a través del reconocimiento de patrones. Estos se clasifican en dos categorías, aplicaciones basadas en estadísticas y en inteligencia artificial. En las aproximaciones basadas en estadística podemos encontrar técnicas tales como PCA (Principal Component Analysis), LVQ (Linear Vector Quantification), redes bayesianas, HMM (Hiden Markov Models), etc. Y en las basadas en inteligencia artificial emplean redes neuronales y sus variantes, tales como PNN (Poligonal Neuronal Network), DWNN (Dynamic Wavelet Neuronal Network), SOM (Self-Organized Feature maps), etc.

El análisis de la supervivencia en mantenimiento predictivo es muy utilizado. Al analizar el tiempo de vida útil residual es común hacer uso del ratio de supervivencia (Hazard Rate), uno de los indicadores más útiles en análisis de supervivencia. Se estima la probabilidad de fallo en un intervalo de tiempo, a través del tratado de datos. Una técnica muy utilizada se conoce como regresión de Cox, o también como modelo de los riesgos proporcionales. Desde que fue desarrollada en 1972 es utilizada en numerosas aplicaciones de medicina, farmacia y mantenimiento, entre otras.

2.2 Técnicas alternativas

2.2.1 Principal Components Analysis, PCA

El Análisis de Componentes Principales, APC en español, es un método algebraico/estadístico de estadísticas multivariables.

Aunque fue creado antes de la segunda guerra mundial, la notable dificultad en los cálculos produjo que no tuviera una amplia aplicación hasta los años sesenta. La popularización del método llegó con la aparición y el desarrollo de los ordenadores.

Este método trata de sintetizar y dar una estructura a la información contenida en una matriz de datos. Ha sido una herramienta ampliamente utilizada en diversas áreas del conocimiento, principalmente cuando se tiene un gran volumen de datos y es necesario conocer cómo están estructurados y cómo se afectan entre sí.

Puede considerarse como una técnica exploratoria de reducción de las dimensiones de una base de datos incorporando variables nominales y ordinales de la misma manera que las numéricas. Al analizar se investigan las relaciones entre las variables originales, entre los casos y entre ambos, además de la relación entre las variables y su nivel de medición. Para variables con relación no lineal pueden especificarse otros niveles de análisis para que se manipulen de manera más efectiva. Por lo tanto, existen dos formas de aplicación.

- **Método basado en correlaciones**: Para datos que no son dimensionalmente homogéneos o que tienen distinto orden de magnitud de las variables medidas.

- **Método basado en las covarianzas**: Para datos dimensionalmente homogéneos y con valores medios similares.

ACP puede ser especialmente útil cuando las variables están muy correlacionadas en el conjunto de datos, tener un conjunto de datos con variables predictivas no correlacionadas complicará la reducción de la dimensionalidad de los datos multivariantes. En estos casos, se puede aplicar, junto a ACP, muchas otras técnicas como "*Forward Feature Selection*" o "*Backward Feature Elimination*".

A la hora de aplicarse en análisis preventivo, es muy útil para reducir altas dimensionalidades de datos a partir de técnicas de aproximación.

Cuando se construye un modelo preventivo es esencial buscar las variables predictivas más importantes. Se analizan tantas variables relevantes como se pueda y luego se enfoca en eliminar las características que no tienen impacto o valor predictivo. También es importante la experiencia de los científicos de datos para saber con qué variables trabajar y qué algoritmos utilizar.

El ACP utiliza un enfoque matemáticamente válido para determinar el subconjunto de los datos que incluyen las características más importantes. Con ello, se conseguirá conjunto de datos más pequeño que el original, pero con un valor predictivo mayor. Para concretar, ayuda a dar sentido a las variables aleatorias identificando el subconjunto de estas que son responsables de la mayor variación con el conjunto de datos original y, además, ayuda a detectar redundancia entre variables.

Al final, para crear un modelo con una eficacia duradera, se debe evaluar cuidadosamente la efectividad de cada variable.

2.2.2 Linear Vector Quantification, LVQ

En computación LVQ es un algoritmo de clasificación basado en prototipos. Puede entenderse como un caso especial de red neuronal artificial que aplica un enfoque basado en la teoría Hebbiana de tipo "*winner-take-all*".

Fue inventado por Teuvo Kohonen. Es un recurso de los mapas auto organizados (SOM), también inventados por el profesor Kohonen, y está relacionado con el gas neuronal y con el algoritmo del vecino más cercano (k-nn).

Un sistema LVQ está representado por prototipos que se definen en un espacio con características de los datos observados. En los algoritmos de entrenamiento "*winner-take-all*" se determina, para cada punto de datos, el prototipo más cercano a la entrada según una medida de distancia dada. A continuación, se adapta la posición de este prototipo, llamado ganador, y con ello se acerca si clasifica correctamente el conjunto de datos y se aleja si lo clasifica incorrectamente.

Entre las ventajas de LVQ destaca que crea prototipos fáciles de interpretar por los expertos en el dominio de aplicación correspondiente.

Al aplicar esta técnica es clave elegir una medida adecuada de distancia o similitud para el entrenamiento y la clasificación. Recientemente, se han desarrollado técnicas que adaptan una medida de distancia parametrizada en el curso del entrenamiento del sistema.

En la evaluación óptima de la estrategia de mantenimiento, la utilización de LVQ estrategia puede representar un buen método con un bajo error de estimación. Al entrenar la red neuronal para calificar entradas, este las analiza y muestra la estrategia óptima a seguir. Al igual que al usar mini-término. Esto es muy útil en el análisis preventivo y permite predecir fallos y tomar decisiones

2.2.3 Redes Bayesianas

Una red bayesiana es una representación de un conjunto de variables aleatorias y sus dependencias condicionales en un grafo acíclico dirigido. Es un modelo que tiene muchas aplicaciones, una de ellas el mantenimiento preventivo.

En el grafo acíclico dirigido de una red bayesiana se representan las variables aleatorias a través de los vértices, o nodos. Las aristas corresponden a una relación de dependencia, *A ->B* (B depende de A). Por último, cada nodo se relaciona con una función de probabilidad que toma como entrada un conjunto de valores de las variables padres del nodo y devuelve la probabilidad de la variable representada por el nodo.

Es una forma compacta y eficiente de codificar la distribución de probabilidad conjunta de varias variables aleatorias. Además, la relación de dependencia de cada variable es condicionalmente independiente de sus no-descendientes dadas sus variables padres, debido a la Propiedad Local de Markov.

Las redes bayesianas aprenden la semántica probabilística utilizando diferentes algoritmos de inferencia. El algoritmo de ingerencia más utilizado es "*Junction Tree Algorithm*". Las probabilidades de una estructura se pueden estimar a partir de los datos utilizando el enfoque de máxima verosimilitud. También pueden actualizarse continuamente a partir de los datos de observación mediante el uso de métodos basados en el gradiente que utilizan sólo la información local derivada de la inferencia.

Para el uso de estas redes en el mantenimiento preventivo, se debe analizar y evaluar los fallos y proponer planes de acción de mantenimiento para incidir en las máquinas.

En el proceso de análisis se deberán identificar las máquinas críticas mediante un análisis cruzado de los indicadores de mantenimiento. A continuación, desarrollar un modelo y una clasificación de prioridades según el análisis anterior. Continuar simulando el comportamiento de las máquinas. Y finalizar proponiendo planes de acción mediante el procedimiento del mantenimiento.

Con este método se pueden mejorar las estrategias de mantenimiento de las máquinas; acorde a la disponibilidad, la mantenibilidad y los costes de mantenimiento.

2.2.4 Replicator Neuronal Network, RNN

Con el avance del Aprendizaje Automático y la Inteligencia Artificial (IA), muchas tareas que requerían de acción humana han sido delegadas a ser realizadas por ordenadores.

RNN fue inicialmente propuesta para identificar anomalías en datos. Propone una red neuronal "*feed-forward*" perceptrón multicapa que funciona como el modelo no supervisado. La funcionalidad de las RNN es reconstruir las entradas dadas en la capa de salida con el error minimizado a través del entrenamiento. Los pesos de la RNN se ajustan para reducir el error cuadrático medio (error medio de reconstrucción) para todos los datos de entrenamiento. Los patrones comunes considerados como "*inliers*" tienen más probabilidades de ser bien reconstruidos, mientras que los anormales considerados como "*outliers"* tienen más probabilidades de ser mal reconstruidos, lo que da lugar a altos errores de reconstrucción.

Esta técnica basada en IA se ha utilizado en muchos ámbitos: detección de fraudes en finanzas, detección de intrusos en la seguridad en la red e incluso detección de cáncer en el diagnóstico médico. En la industria pueden utilizarse para detectar e identificar anomalías o valores atípicos considerados como el potencial fallo de las máquinas en funcionamiento.

La RNN podría utilizarse para identificar valores atípicos en los datos sin utilizar la etiqueta de clase con una alta precisión en el mantenimiento. Para construir un modelo de predicción de datos de mantenimiento predictivo, se requieren datos muestreados para entrenar la RNN. El modelo desarrollado (Outlier Factor) da una medida cualitativa de la perificidad basada en el error de reconstrucción, cuanto mayor sea el error de reconstrucción, mayor será la probabilidad de mal funcionamiento.

Con un desarrollo adecuado, se puede convertir una red neuronal profunda típica con tamaño reducido y en ejecutarla en un dispositivo perimetral, lo que permitiría el desarrollo de una inteligencia artificial integrada. Llevar esta inteligencia artificial al sistema supondría considerables beneficios como: eficiencia de costes, energética y reducción del consumo.

2.2.5 Dynamic Wavelet Neuronal Network, DWNN

DWNN pertenece a una nueva clase de redes neuronales con capacidades únicas para abordar problemas de identificación y clasificación. "*Wavelets*" son una clase de elementos básicos con oscilaciones de duración finita con apariencia de pequeñas ondas. La naturaleza autosimilar y de resolución múltiple de estas ofrece una ventana al análisis de señales físicas e imágenes. Por otro lado, las redes neuronales artificiales constituyen una potente clase de aproximaciones de funciones no lineales para la estimación sin modelo. Al introducir una red neuronal artificial se permite aprovechar de forma coherente el terreno común de estas dos tecnologías.

En un sistema estático WNN se establece una relación estática entre sus entradas y salidas. Las redes neuronales dinámicas o recurrentes son necesarias para moldear la evolución temporal de los sistemas dinámicos. Las señales en una configuración de red de este tipo pueden fluir hacia delante y hacia atrás (desde la salida hasta los nodos de entrada). Sin embargo, un WNN dinámico multi-resolución, utiliza la transformación de wavelet discreta y redes neuronales recurrentes que forman modelos no lineales para la predicción.

Al usar la DWNN en mantenimiento preventivo, se extraen características de la máquina periódicamente para que se procese un pronóstico. Las características son procesadas de forma dinámica y, a continuación, fusionada en un vector dependiente del tiempo. La selección de características se basa en criterios que distinguen una firma de fallo de las condiciones de funcionamiento normales y un modo de fallo concreto de otro. También pueden incluirse otros criterios como el coste computacional.

Antes de la implementación y el uso en real, la DWNN debe ser entrenada y validada. Para el entrenamiento se pueden utilizar algoritmos tales como el de Retropropagación o el Genético. Una vez entrenado, puede actuar en real junto a la medida de pronóstico de tiempo hasta el fallo (*Time-To-Failure*).

A la hora de elaborar el pronóstico se utilizan datos del diagnóstico, por lo que la fidelidad y precisión del diagnosticador tienen un impacto directo en la fiabilidad del pronosticador.

2.2.6 Self-Organized Feature Maps, SOM

SOM es un tipo de red neuronal artificial, entrenada utilizando aprendizaje automático no supervisado. Es utilizada para producir una representación de dimensionalidad reducida (normalmente una o dos dimensiones) de un conjunto de datos de dimensionalidad superior a la vez que mantiene las propiedades topológicas de los datos.

Como se dijo en el apartado 2.2.2, fue inventado por Teuvo Kohonen y utiliza LVQ como recurso.

Al configurar el SOM, se colocan las neuronas de los nodos de una red unidimensional o bidimensional. También son posibles mapas de mayor dimensión, aunque no son comunes. Las neuronas se ajustan selectivamente a varios patrones de entrada (estímulos) o clases de patrones de entrada durante el aprendizaje.

Esta clase de sistema no supervisado se basa en el aprendizaje competitivo, en vez de aprendizaje por corrección de errores. En él, las neuronas de salida compiten entre sí para ser activadas, con el resultado de que solo una es activada en cada momento. La neurona activa se le conoce como neurona ganadora, siguiendo el método "*winner-take-all*". Esta competencia puede ser introducida teniendo vías de retroalimentación negativa entre las neuronas. Como resultado, las neuronas ganadoras se ven obligadas a organizarse a sí mismas, por ello se llaman "*Self-Organized Feature Maps*" algo que en español podría traducirse como mapas autoorganizados. Las ubicaciones de las neuronas que han sido sintonizadas así (ganadoras) se ordenan y se crea un sistema de coordenadas significativo para las características de entrada. Así es como el SOM forma el mapa topográfico necesario de los patrones de entrada.

El proceso SOM se puede dividir en cuatro componentes principales:

- **Inicialización**: Todos los pesos se inicializan con pequeños valores aleatorios.
- **Competencia**: Para cada patrón de entrada, las neuronas calculan sus respectivos valores de una función discriminante que proporciona la base para la competencia. Se elige la neurona con el valor de discriminación más pequeño como ganadora.
- **Cooperación**: La neurona ganadora determina la ubicación espacial de una vecindad topológica de neuronas activadas, creando la base de para la cooperación entre neuronas vecinas.
- **Adaptación**: Las neuronas activadas reducen sus valores de la función de discriminación en relación con el patrón de entada mediante un ajuste de pesos de conexiones asociadas.

Este proceso se puede interpretar como una generalización no lineal del Análisis de Componentes Principales, del cual se habló en el apartado 2.2.1.

2.3 Propuesta

Para lograr un análisis correcto de las máquinas se precisa de un entrenamiento previo de los algoritmos de mantenimiento preventivo empleados. Por desgracia, entrenar estos algoritmos con datos reales supondría un coste elevado de tiempo y recursos.

Para conseguir los mismos resultados que un entrenamiento sobre máquinas reales, pero de forma rápida y fácil, se ha desarrollado una herramienta que simula los datos que generaría el análisis de las distintas máquinas. En esta herramienta, se podrá seleccionar fácilmente parámetros que permitirán cambiar la situación a simular, tales como la frecuencia de escaneo que se realizaría sobre la maquinaria, el número de muestras o el momento en el que se presenta el fallo. También se permitirá cambiar el tipo de análisis que se va a ejecutar y, tras esto, se podrán ver los resultados de forma gráfica y fácil y, si así se desea, guardarlos en formato Excel para que luego sean utilizados en el entrenamiento.

En el desarrollo de esta herramienta se ha especificado que deberá ser utilizado el lenguaje de programación *R*. Teniendo esto en cuenta, en el diseño de la interfaz se ha optado por utilizar el paquete de *R* llamado Shiny y, dentro de las posibilidades que ofrece Shiny, se ha extendido la personalización del aspecto a través del uso de CSS.

3 Análisis

En este apartado se realizará un análisis exhaustivo para aclarar y precisar el producto final.

3.1 Propósitos

Como se ha dicho anteriormente, el propósito principal del proyecto es crear una herramienta para lanzar simulaciones, que recrearán los datos recogidos de las máquinas de una cadena de montaje, para entrenar un sistema predictivo de fallos.

Para ello, se deberá crear una interfaz desde la que se pueda elegir entre varios tipos de simulación y unos parámetros. Además, deberá mostrarse gráficamente los datos de cada ejecución y permitir guardar fácilmente estos en formato Excel.

3.2 Programa principal e interfaz

En el desarrollo de la interfaz, se ha ejemplificado una idea del resultado deseado a través de una aplicación actual utilizada para análisis de Ford.

Imagen 1: Ejemplo de interfaz deseada

Aunque la imagen número uno es un ejemplo del objetivo, la interfaz a desarrollar cambiará para adaptarse mejor al objetivo del proyecto.

En el resultado final se eliminarán elementos innecesarios como la pieza del equipo a la que hacen referencia los datos, ya que la ejecución va a ser general y es un dato que no importa. Otro apartado eliminado será el recuadro '*Char Options*' ya que son unas opciones que no se utilizarán.

Se ejecutará un cambio al sector de '*Data Statistics*' que pasará a mostrar un diagrama de barras que representará el número de veces que una muestra es cada valor simulado.

Por último, por lo incómodo que resulta tener que cambiar a una pestaña de configuración a establecer los parámetros a ejecutar y el tipo de simulación. Se ha propuesto cambiar esa selección a un panel o apartado en la parte izquierda desde el que se pueda configurar todo sin tener que cambiar la vista.

A continuación, en la imagen número dos se muestra uno de los primeros conceptos de la interfaz, aunque algunas de las ideas plasmadas en la siguiente imagen sufrieron cambios a medida que se avanzaba en el desarrollo y otras ya habían evolucionado de la idea inicial.

Imagen 2: Concepto de interfaz inicial

Por último, la imagen número tres es un concepto de la idea final para la interfaz. A partir de este solo se realizaron pequeños cambios.

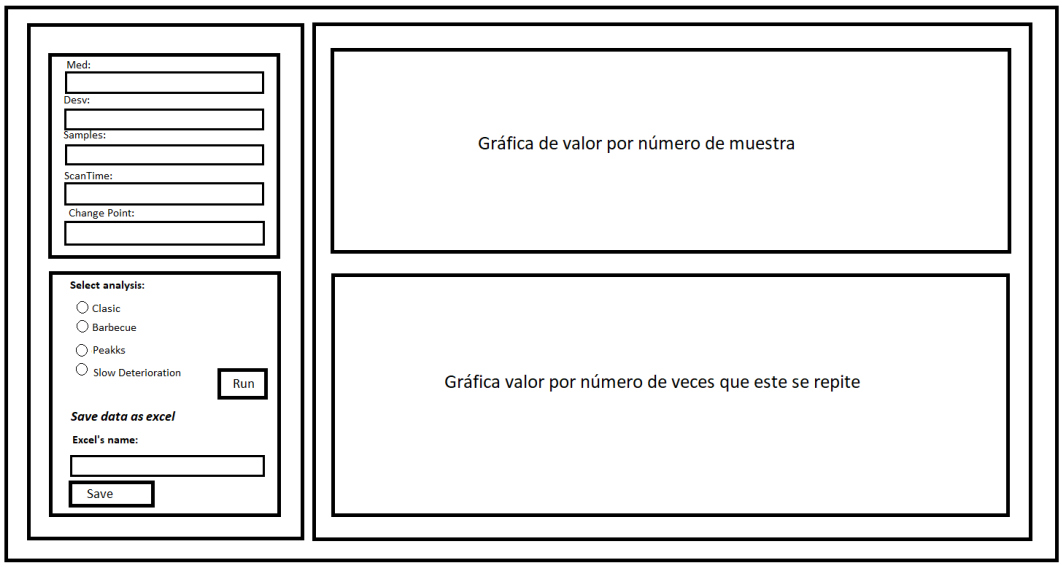

Imagen 3: Concepto de interfaz final

3.3 Parámetros

- **Med:** Este parámetro hace referencia a la media de valores que se detectarían de la máquina normalmente, antes de que ocurra ningún fallo. Se recuerda que este valor puede hacer referencia a nivel de ruido, vibración, etc.

- **Desv:** Este valor se refiere a la desviación típica. Es una medida que cuantifica la dispersión del conjunto de datos. Si este valor es bajo los datos aparecerán más agrupados y se irán dispersando cuanto más aumente.

- **Samples:** El número de "muestras" literalmente en español. Indica el número de valores que van a calcularse.

- **ScanTime:** Este parámetro hace referencia al tiempo que transcurriría entre la detección de cada muestra en maquinaria real. También podríamos referirnos a él como la frecuencia de medida de datos.

- **Change Point (%):** También conocido como punto de cambio en español. Como se comentó anteriormente en el apartado 2.1.2.b, este valor indica un cambio abrupto de la medida que se está realizando sobre la máquina. Indica que hay algo anómalo con la maquinaria y es significativo de que la vida útil de algún componente está llegando a su fin. Este valor representa en qué punto de las muestras se simula (p. ej. si se establece a 30 representa que se simulará este cambio abrupto tras el 30% de las muestras).

3.4 Tipos de simulación

Como ya se ha comentado anteriormente, este proyecto consta de un programa principal desde el que se pueden ejecutar varios tipos de simulaciones. Cada simulación recrea los datos que devolvería la maquinaria que está sufriendo un tipo distinto de error. En este apartado se analizará los cuatro tipos de algoritmos entre los que se podrá elegir para ejecutar la simulación.

- **Clasic:** Esta simulación representa los datos que devolvería maquinaria que, por algún fallo tras pasar el punto de cambio, empieza detectar valores superiores a la media. Tras este cambio la media de los nuevos valores es más elevada que la anterior.

- **Barbecue:** Esta simulación representa un cambio en características de la máquina tales como la velocidad o similares. Produce un cambio abrupto que puede ser positivo o negativo.

- **Peakks:** Esta simulación muestra datos que aparecen frecuentemente en el mini-término pero de amplitud y frecuencia aleatorios. Gráficamente se pueden observar datos muy agrupados con algunos considerablemente por encima de la media. Es un problema típico de las abrazaderas.

- **SlowDet:** Esta simulación produce los datos que devolvería maquinaria que sufre un deterioro lento. Se puede detectar un cambio lento en la media de los datos.

3.5 Soluciones alternativas posibles

[1]3.5.1 R vs MATLAB

Aunque uno de los requisitos del proyecto impuesto por Ford es que el desarrollo fuera en R, en este apartado se comparará el lenguaje de programación R con MATLAB otro potente lenguaje de programación. El objetivo de este punto es describir por qué R es una alternativa más que acertada.

Para realizar una breve introducción de estos lenguajes. R es un programa de código abierto popular y potente para computación estadística y gráficos. Por otro lado, MATLAB es un lenguaje diseñado para ingenieros y científicos muy utilizado en computación matemática. Para compararlos vamos a analizar cinco aspectos de estos lenguajes:

- **Facilidad de aprendizaje:** R es conocido por tener una curva de aprendizaje [2]difícil. Se accede a toda su capacidad a través de la programación y, hasta hace relativamente poco, no había una interfaz de usuario para que los no programadores ejecutaran análisis. Sin embargo, con la aparición de RCommander y RStudio y la introducción de interfaces de usuario por estos, se ha ayudado mucho a la comunidad de desarrollo en R. Por otro lado, MATLAB es un lenguaje con una sintaxis simple y consistente que resulta fácil de aprender y recordar incluso para no programadores. En este sentido, se podría decir que MATLAB supera a R. Como en el desarrollo de este proyecto ya se disponía de conocimientos de programación para mi no ha sido un problema destacable.

- **Coste:** R es un producto de código abierto y, por lo tanto, gratuito. En contra a esto, MATLAB requiere de una licencia con costes variables dependiendo del tipo de uso. Indiscutiblemente este punto es a favor de R.

- **-Rendimiento:** En este sentido, al realizar tareas de computación, estadísticas y aprendizaje automático MATLAB es más rápido que R. Requiere de alguien experto en el uso de R para competir con MATLAB, aunque en esta situación se pueden lograr resultados más rápidos y mejorar el rendimiento. En este sentido se podría decir que la balanza está ligeramente a favor de MATLAB.

- **Apoyo y documentación:** Teniendo en cuenta que R es un lenguaje de código abierto, tiene una amplia comunidad de desarrolladores que ofrecen apoyo y documentación. Aunque, en lo que se refiere a documentación para MATLAB, *MathWorks* ofrece un apoyo excepcional. Aporta documentación, con cientos de ejemplos, disponible online y a través de la aplicación de escritorio. En este sentido, ambas opciones disponen de buenas opciones fácilmente al alcance del usuario.

[1] MATLAB vs R | Learn The Top 7 Important Differeces (educba.com)
R vs MATLAB in 2021: Comparison, Features & Applications (hackr.io)

- **Integración con otros lenguajes:** Empezando por MATLAB, este ofrece integración con otros lenguajes de programación en dos sentidos, se pueden hacer llamadas a MATLAB desde otro lenguaje y desde MATLAB se puede hacer llamadas a librerías en otros lenguajes. Además, el código puede ser convertido a código C/C++. De forma similar, gracias al paquete '*knitr*' de R, este da soporte a multitud de lenguajes tales como Python, Julia, C++, SQL, etc.

- **Uso habitual:** R es el lenguaje preferido para análisis de datos en entornos industriales, mientras que MATLAB se utiliza principalmente en ámbitos universitarios.

Para concluir, ambos lenguajes presentan muchas similitudes y unas diferencias que pueden tener más valor o menos dependiendo de cada individuo. En el caso de este proyecto no tener que disponer de una licencia a la hora de desarrollarlo o de utilizarlo en un futuro es un argumento decisivo para el uso de R, sobretodo teniendo en cuenta que no hay contras significativos.

3.5.2 RStudio vs Commander

Como se ha comentado en el apartado anterior, es difícil empezar a programar en R y, para ayudar en este proceso, se han creado RStudio y RCommander.

RStudio es un entorno de desarrollo integrado (*IDE*) creado para apoyar el desarrollo de código R. Usa el entorno gráfico del ordenador para facilitar interacciones con R. RStudio dispone de dos opciones de ejecución, una versión de escritorio para Linux, Windows y Mac o una versión server para Linux.

RCommander, o Rcmdr, es una interfaz gráfica de usuario para análisis estadístico situada sobre R. Aunque gráficamente no muestra mucho, consiste en una ventana que muestra el código del script, otra para la salida y una para mensajes de Windows. R se ejecuta en otra ventana y puede ser utilizado directamente en cualquier momento. Fue creado con el propósito de acceder a los métodos estadísticos más esenciales de forma simple, es más conveniente para usuarios que han usado SPSS, SAS o Stata (programas que pueden resultar similares en analíticas y estadísticas de datos). No proporciona acceso directo a la línea de comandos de R pero muestra el código que se está ejecutando.

Teniendo en cuenta las características de ambos. RStudio ha sido la opción elegida para el desarrollo del proyecto. Ofrece unas características más útiles y familiares para el método seguido al desarrollar. Aunque RCommander pueda resultar muy útil en estudio de datos, RStudio se adapta mejor al objetivo del proyecto.

4 Diseño

En este apartado se realizará una explicación más detallada del proyecto realizado empezando con la presentación y a continuación la lógica.

4.1 Entorno gráfico

Para la presentación gráfica del simulador se realizaron unos bocetos en papel como base a las primeras ideas presentadas, también se pasaron a la aplicación Paint y fueron modificados a lo largo del desarrollo. Se optó por esta herramienta porque para plasmar y guardar las ideas como bocetos es sencilla, gratuita y está instalada de forma predeterminada en los equipos de Windows. Anteriormente, en el apartado 3.2, se detalló estos pasos para el desarrollo de la interfaz.

En cuanto a la hora de desarrollar una interfaz gráfica de usuario en R, las investigaciones mostraban que R no es muy amigable a la hora de desarrollar una interfaz. Tras investigar más, se determinó que shiny podría ayudar en este proceso. Shiny es un paquete de R que agiliza y simplifica el desarrollo de interfaces en R y, unido a CSS, permite personalizar el aspecto de la interfaz.

4.2 Lógica

A continuación, se explicarán las funcionalidades del proyecto:

Ejecutar una simulación: El usuario podrá configurar mediante un menú qué simulación quiere ejecutar y con qué parámetros. El usuario puede determinar cada parámetro a través de los cuadrados de texto editables bajo la etiqueta correspondiente a cada uno. Paralelamente, a través de unas casillas de marcaje puede elegir uno de los cuatro análisis desarrollados para que este se ejecute cuando pulse el botón "*Run*". Ejecutar la simulación mostrará los resultados como dos gráficas en el panel derecho.

Guardar resultados: El usuario podrá guardar los resultados como un Excel pulsando el botón "*Save Excel*". Antes de guardarlo puede elegir en unas casillas de selección si quiere que el nombre del Excel se determine automáticamente o si desea usar un nombre especificado por el usuario. Si elige automáticamente, como está por defecto, se usará la fecha y hora actuales para nombrar el archivo. Si, por el contrario, quiere que se utilice un nombre escrito por el usuario, podrá determinar un nombre en el cuadro de texto situado bajo la etiqueta "*EXCEL'S NAME:"*. Una vez guardados, los datos se guardan en una carpeta llamada "Datos" en la ubicación del programa.

5 Desarrollo

Como se ha especificado anteriormente, el objetivo principal de este proyecto es desarrollar un método para simular los datos que se obtendrían, como resultado del análisis de una máquina en una línea de producción, a través de mini-término.

Se pretende con esto dar apoyo a un proyecto académico innovador y en desarrollo. Un proyecto que pretende determinar la patología que sufre una máquina cualquiera, de una cadena de montaje, así como el tiempo de vida restante que le queda hasta presentar un fallo en la línea de producción. Con ello, supondrá un ahorro económico enorme a la compañía, por su sistema predictivo, logrando minimizar los paros de línea y realizar una programación óptima para las reparaciones de los componentes deteriorados.

El método de simulación de los datos se ha desarrollado en R y, además, se ha utilizado el paquete shiny para la realización de la interfaz gráfica de usuario.

5.1 Lenguaje de programación R

R es un software de licencia gratuita, un lenguaje de computación formal diseñado para manipulación de datos y análisis estadístico. Puede ser considerado como un lenguaje orientado a objetos ya que el resultado de la evaluación de cada función genera un objeto, con atributos. Consiste en un lenguaje y entorno de ejecución con gráficos, un depurador (*debugger*), acceso a ciertas funciones del sistema y la habilidad de ejecutar programas almacenados en scrits.

El lenguaje R fue creado en 1996 por Robert Gentleman y Ross Ihaka, de la Universidad de Auckland de Nueva Zelanda, como una iniciativa de desarrollo basándose en el lenguaje S.

Se han comentado más detenidamente aspectos del lenguaje de programación R en el apartado 3.5.1.

5.2 RStudio

Desde el 28 de febrero de 2011, R cuenta oficialmente con un entorno de desarrollo integrado (*IDE*) llamado RStudio. Como se comentó previamente en el apartado 3.5.1, RStudio supone una gran ayuda para el desarrollo en R y permite contar con una interacción más fluida con el programa R. Incluye una consola, una interfaz de usuario, editor de sintaxis durante la ejecución del código y herramientas para el trazado, la depuración y la gestión del espacio de trabajo.

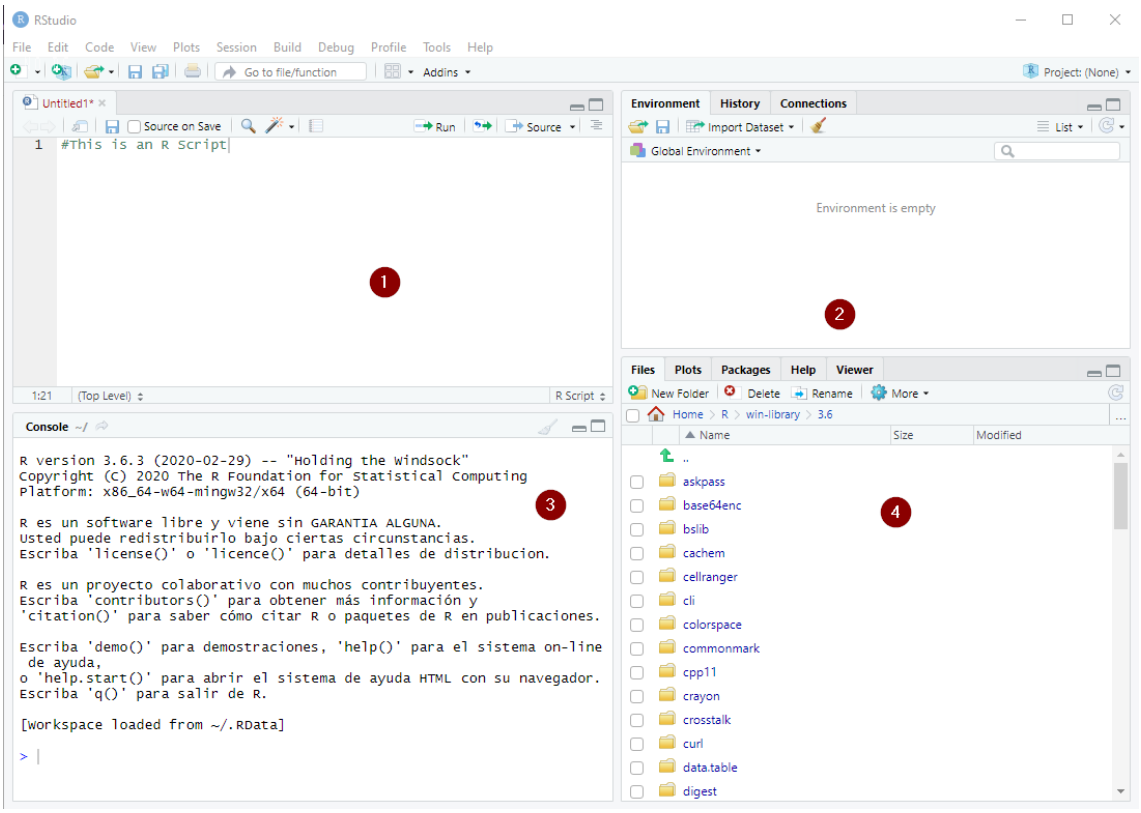

Imagen 4: Entorno RStudio

Como se puede apreciar en la imagen cuatro, RStudio cuenta de cuatro ventanas, además de la barra de opciones de la barra superior.

La primera ventana es el editor de sintaxis, el lugar desde el que editar el script que posteriormente se ejecutará.

La segunda ventana es el entorno de trabajo del programa. En este lugar se muestra el conjunto de datos y objetos que se almacenan al ejecutar diferentes análisis (resultados, variables, funciones, gráficos, etc.)

La tercera ventana es la consola. Corresponde a lo que sería el software R en su versión básica.

La cuarta ventana tiene varias pestañas:

- *Files* es directorio de archivos.

- *Plots* es donde se muestran por defecto los gráficos generados.
- Packages muestra los paquetes descargados y almacenados en el disco duro, desde esta pestaña se puede gestionar su instalación o actualización.

- *Help* permite consultar recursos sobre el aprendizaje de R, durante el desarrollo permite consultar fácilmente información sobre las funciones proporcionadas que se están utilizando. Además, cuenta con manuales, referencias y más material que hacen mucho más sencillo trabajar con R.

- *Viewer* muestra los resultados al construir reportes mediante funcionalidades como *rmarkdown*.

5.3 Paquetes utilizados

La funcionalidad de R consta de paquetes modulares. La base de R contiene paquetes básicos para su ejecución y funciones básicas del lenguaje para leer y manipular datos, funciones gráficas y funciones estadísticas. En adición a esto, la comunidad ofrece otros paquetes que son de ayuda a la hora de programar. Algunos de los paquetes más utilizados según su función son:

- **Carga de datos:** SQLdf, RODBC, RPostgresSQL, RSQLite, foreign, readxl.
- **Informes:** shiny, knitr, xtable.
- **Manipulación de datos:** plyr, lubridate, reshape2, stringr.
- **Modelado:** caret, car, randomforest, qcc, zoo, forecast.
- **Visuacización de datos:** ggplot2, rgl.

En el desarrollo de este trabajo se han utilizado algunos de estos paquetes; concretamente readxl, shiny, stringr y ggplot2. Y, aunque son excelentes paquetes a la hora de agilizar el desarrollo de múltiples proyectos, solo se desarrollará shiny.

Hablando brevemente del resto de paquetes utilizados. Readxl es un paquete que ofrece instrucciones para leer y escribir información en archivos Excel, Stringr simplifica el trabajo con *Strings* y, para finalizar, ggplot2 es una herramienta diseñada para mostrar datos de forma gráfica.

Por otro lado, ya que un requisito fundamental del proyecto era poder interactuar con una interfaz, cabe destacar a Shiny sobre el resto de paquetes. Este componente permite crear aplicaciones web interactivas y reactivas que pueden ser abiertas en un ordenador, una tablet o un móvil. El desarrollo de cualquier aplicación shiny se basa en dos partes: la interfaz del usuario que recibe los inputs y muestra los outputs y un script para realizar los cálculos necesarios, conocida como server.

5.4 Desarrollo de la interfaz

El desarrollo de la interfaz de usuario se ha basado en el concepto de rejillas (conocido más comúnmente como *Grids*). Usando este principio se han añadido distintos elementos proporcionados por shiny de forma alineada, consistente y manteniendo un código estructurado y limpio.

La distribución se basa en dos paneles principales, uno lateral y otro principal.

En el panel lateral se pueden diferenciar dos partes, un panel superior y otro inferior. En el panel superior se han colocado cuadros de texto, en los que escribir los parámetros con los que se calcularán las simulaciones y se representarán las gráficas, junto a las etiquetas de qué valor hacen referencia. En el panel inferior se puede observar un selector de simulación, que permite elegir el deseado, seleccionando su casilla y un botón llamado *Run* para calcular y mostrar los resultados de la simulación. Bajo esto, se encuentra otro selector para elegir si, al pulsar el botón *Save Excel*, los resultados se guardarán en un archivo '.csv' con un nombre generado o con un nombre introducido en el cuadro de texto que se observa sobre el botón.

En el panel principal, se muestran dos gráficas que hacen referencia a los resultados de la simulación. La gráfica superior referencia el valor simulado y su número de muestra. Las muestras [1, 2..n] representan además el avance del tiempo sobre la máquina simulada, habiendo un tiempo definido por el parámetro ScanTime entre cada muestra y su siguiente.

5.5 Algoritmos

A la hora de desarrollar los algoritmos para la ejecución de cada simulación se siguió una guía desarrollada en MATLAB y se adaptó a R.

Cada algoritmo ha tenido que ser entendido, adaptado y comprobado. Más tarde, se han realizado unas modificaciones para añadir aspectos, como parametrizar el *change point* y poder configurarlo antes de la ejecución.

5.6 Comportamiento de los botones

Para especificar el comportamiento de los botones comentados en el apartado anterior, se han realizado unos diagramas de flujo que representan el comportamiento del programa tras pulsarlos.

El botón "*Run*":

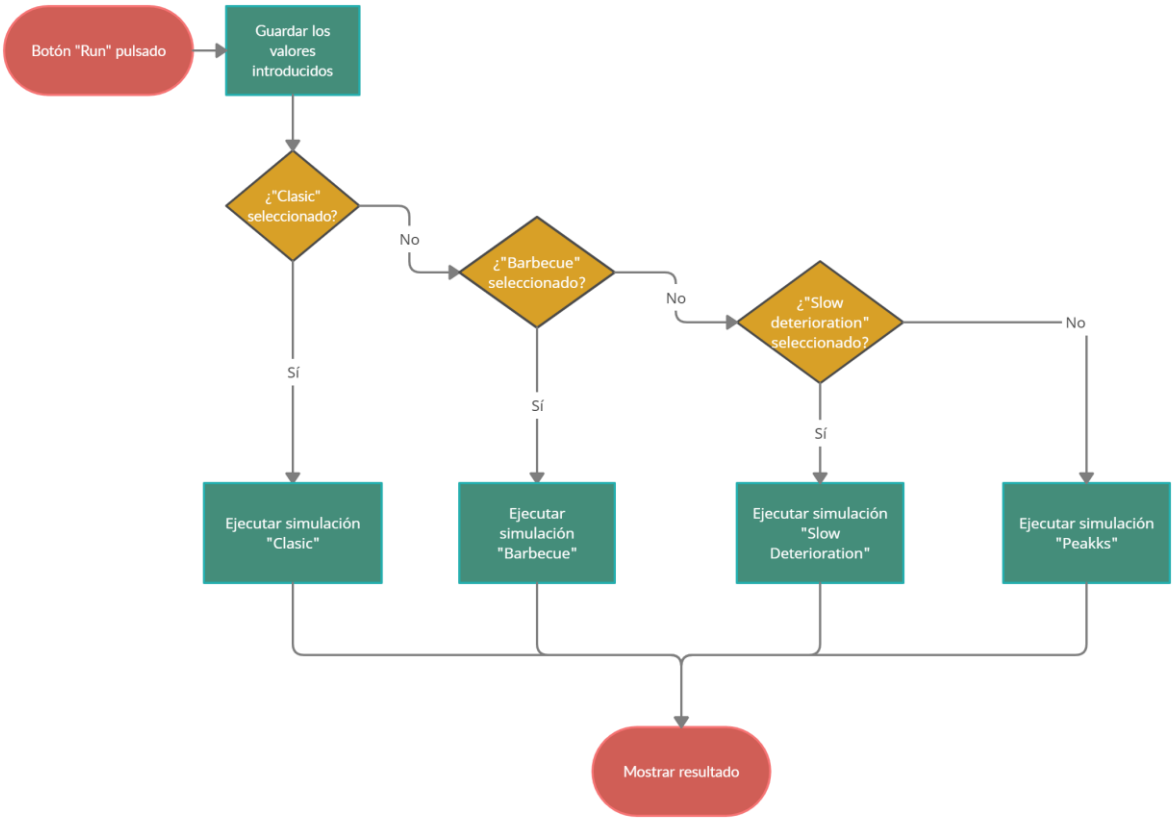

Imagen 5: Comportamiento botón Run

El botón “*Save Excel*”:

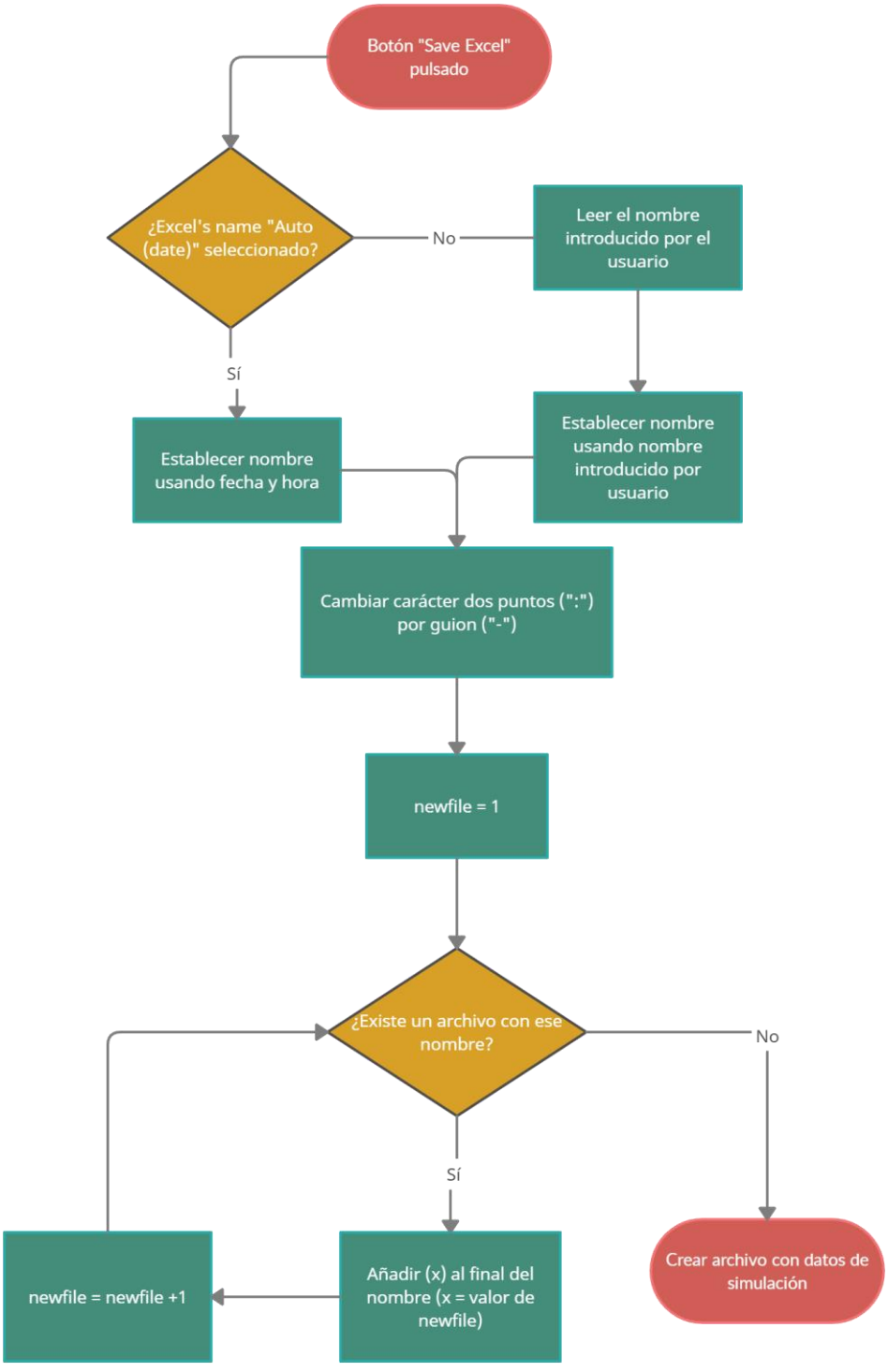

Imagen 6: Comportamiento botón Save Excel

[3]5.7 CSS

A la hora de cambiar el aspecto de una aplicación desarrollada con shiny, hay varias formas de aplicarlo, pero se pueden calificar como dos principales:

- **Inline CSS**: Son las instrucciones de CSS escritas utilizando las opciones que ofrece shiny entre las instrucciones del código de la interfaz de usuario.
- **File-based**: Las instrucciones para el estilo están escritas en un archivo de tipo .css separado y la aplicación apunta a él para cargar el aspecto.

En el desarrollo del aspecto del proyecto, se ha abordado con un desarrollo mixto. Aunque la opción de crear un archivo separado en el que se especifica el aspecto está presente, se han introducido algunos cambios por código junto a la interfaz. Como la jerarquía utilizada en el archivo es mayor que la utilizada en el código, los cambios introducidos en el código de la interfaz serán visibles si no se consigue cargar la interfaz, permitiendo que si se elimina el archivo siga teniendo un aspecto amigable.

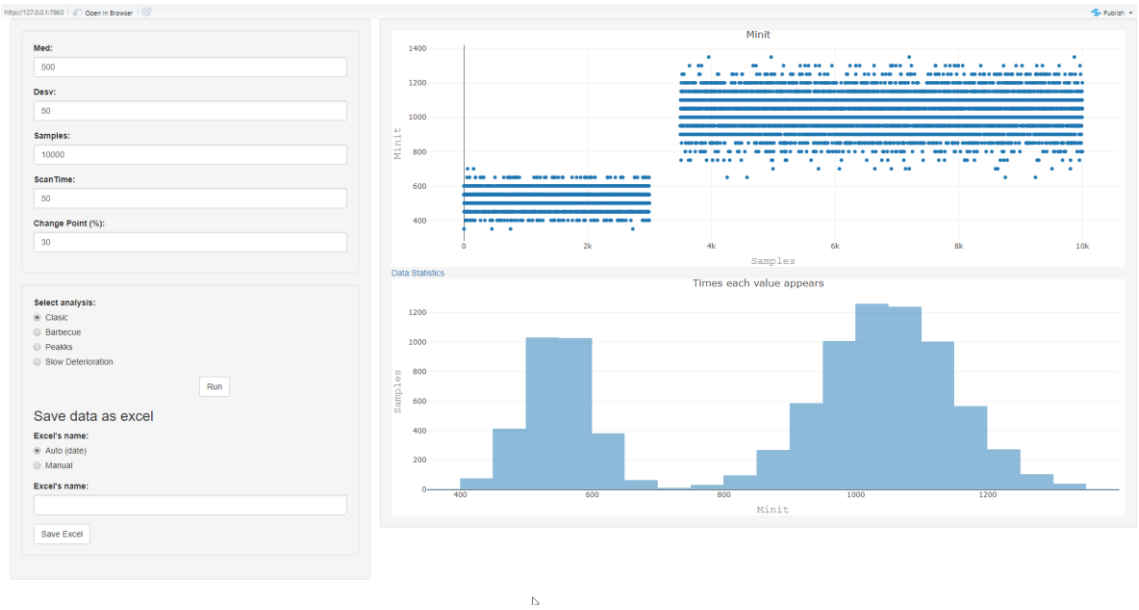

Imagen 7: Programa sin cambios de CSS

[3] https://shiny.rstudio.com/articles/css.html

Imagen 8: Programa con cambios de CSS

En la imagen 6, por otro lado, se puede ver los cambios realizados al cargar el archivo style.css.

6 Implantación

A la hora de empezar a utilizar la herramienta es necesario preparar antes el equipo. Para ello, deberá de disponerse de todos los archivos del programa y se deberán descargar las librerías citadas a continuación o ejecutar el script preparado para instalarlas todas a la vez. En este punto se explicará cómo.

6.1 Preparar sistema

La carpeta llama "*R Miniterm Sym*" contiene los ficheros necesarios para esta aplicación. Los archivos y carpetas necesarios para el uso de esta aplicación son:

- **Main.R**: Es el archivo principal de la ejecución del programa. Ejecuta el entorno, carga todas las librerías y llama a los archivos que contienen las simulaciones, inicializándolas. Carga la interfaz y sus funcionalidades.
- **Barbacue.R, Clasic.R, Peakks.R, SlowDet.R**: Cada uno contiene la simulación correspondiente a su nombre.
- **Librerias**.R: Es un archivo que sirve como script sencillo para instalar las librerías que utiliza el simulador.
- **Datos**: Es un fichero que se crea automáticamente en el que se guardan los resultados de la simulación cuando el usuario pulsa el botón correspondiente.
- **www**: Es un fichero que contiene el archivo style.css.
- **style.css**: Es el archivo que contiene el código CSS para cargar el aspecto de la interfaz.

6.2 Librerías

Esta herramienta ha sido desarrollada utilizando las siguientes librerías R: shiny, plotly, ggplot2, stringr y readxl. Por lo tanto, para su funcionamiento es necesario que estas se encuentren instaladas en el sistema de R.

Para instalar una librería en R hay que introducir el comando "*install.package(XXXX)*", siendo "*XXXX*" el paquete a instalar, en la consola de R. Puede accederse a ésta ejecutando R o desde la consola de R integrada en RStudio como se indicó anteriormente (en el apartado 5.2).

Si no se desea instalar las librerías manualmente se puede ejecutar el script "Librerias.R".

R permite la opción de escribir en el código del programa la instalación de librerías. Sin embargo, ejecutar el comando para instalar una librería ya instalada la actualizará, si cada vez que se ejecutara el simulador R actualizara las librerías provocaría un aumento no deseado en el tiempo de ejecución.

7 Pruebas

Después de haber completado la implementación, se debe probar distintas ejecuciones con distintas combinaciones de parámetros para, así, comprobar que el objetivo se ha alcanzado correctamente, y si no es así, solucionar los problemas que puedan haber aparecido durante las pruebas.

En este apartado se detallarán las pruebas realizadas así como los problemas que han aparecido a raíz de estas.

7.1 Combinaciones

Durante las pruebas relacionadas al desarrollo del código de los algoritmos, se han ejecutado distintas variaciones en la configuración de entrada y se han contrastado los resultados con ejecuciones de los algoritmos desarrolladas en MATLAB para comprobar que el código desarrollado en R estaba funcionando correctamente.

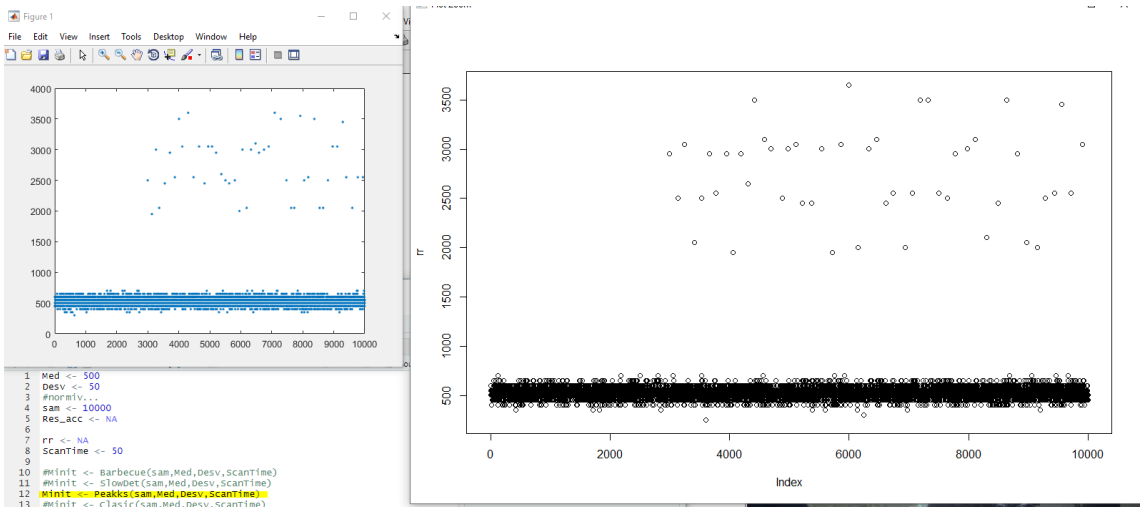

Imagen 9: Comparativa algoritmo Peakks

Como se puede apreciar en la imagen 7, los datos generados con ambos métodos comparten unas características similares.

7.2 Análisis de tiempos

Para analizar los tiempos de ejecución se ha ejecutado el programa con unos parámetros y un número habitual de muestras a generar. Luego se ha ido aumentando el número de muestras hasta una cantidad muy elevada para ver cómo los tiempos van aumentando.

Simulación Muestras	Classic	Barbecue	Peakks	Slow Deterioration
10.000	1" 55 ms	1" 30 ms	1" 04 ms	1" 57 ms
50.000	23" 36 ms	3" 49 ms	9" 10 ms	8' 51 ms
100.000	41"44 ms.	17" 07 ms	1' 9"	10' 23 ms
500.000	12' 03"	18' 34 ms	6' 44"	16" 36'

Tabla 1: Tiempos de ejecución con distintas muestras

Como se puede apreciar, existen algunas variaciones con el tiempo entre los algoritmos. Aunque los tiempos de ejecución aumenten mucho, normalmente la cantidad de datos generada está alrededor de los 10.000, por lo tanto, no debería ser un problema.

Al realizar un análisis más profundo de en qué se gasta el tiempo, se ha descubierto que crear los gráficos con los datos (sobre todo cuando son muchos datos) ralentiza mucho el programa. Para volúmenes de datos normales no es un problema, pero cuando se elevan a muchas muestras podría llegar a serlo. Por la investigación que se ha realizado no hay opciones que presenten resultados mucho mejores en R, para solucionar este problema habría que esperar una actualización o desarrollo de una alternativa que permita generar los gráficos de una manera más rápida.

7.3 Nombres de archivo

Estas pruebas han consistido en probar que, a la hora de generar archivos y nombrarlos, las medidas desarrolladas funcionaban correctamente. Se han generado:

- Distintos tipos de simulación con nombres automáticos.
- Archivos con nombres iguales para comprobar que añadía el segundo con un número entre paréntesis, y luego otra vez para ver que el número aumentaba con cada archivo.
- Nombres con ":" (que es un carácter que Windows no permite en sus nombres) y el nombre ha sido adaptado correctamente.

8 Conclusiones

Con este apartado se concluye el proyecto. Al hacer repaso de los objetivos que han sido planteados, solo queda unir el desarrollo de este proyecto con lo desarrollado en el proyecto de investigación de mini-término.

En este proyecto se ha desarrollado un método de ejecutar cuatro simulaciones, configurarlas a través de una interfaz y ver los resultados de forma gráfica, permitiendo generar un archivo con estos resultados si así se desea.

Empezar a utilizar estas simulaciones supondrá poder generar en poco tiempo una cantidad de datos que en realidad se tardaría mucho más en obtener. Con esto, se cumplirá el objetivo que permitirá entrenar los algoritmos para aumentar la eficiencia del mantenimiento preventivo de la maquinaria.

No obstante, es posible que en un futuro se desarrollen y haya que implementar más técnicas de simulación. Es probable que, tras ser utilizarlas dentro de un tiempo indefinido o por cambios en la maquinaria y sus componentes, se concluya que alguna debe ser adaptada o cambiada.

Por lo tanto, este proyecto deberá mantenerse actualizado a los últimos desarrollos que se realicen en el uso de mini-término para Ford. Es importante mantener acorde a la investigación y desarrollo para conseguir resultados fiables y útiles.

Para finalizar, hay que recordar que el proyecto al que este proyecto da respaldo es un proyecto innovador en desarrollo. Por ello, se espera que el desarrollo de este proyecto sea un gran apoyo y puedan continuar desarrollándose e innovando.

9 Referencias

MOTIVACIÓN

Eduardo García Magraner y Nicolás Montés Sánchez; "Mini-term, a novel paradigm for fault detection"; ScienceDirect, 2019.

CRÍTICA AL ESTADO DEL ARTE

Nicolás Montés Sánchez; "Big Data basado en los sub-tiempos de ciclo técnico para la mejora del rendimiento de las líneas de fabricación. (Mini-términos 4.0)"; Propuesta Científica, 2018.

2.1.2 MANTENIMIENTO PREVENTIVO/PREDICTIVO (PM)

Peng, Y., Dong, M. & Zuo, M.J. Current status of machine prognostics in condition-based maintenance: a review. Int J Adv Manuf Technol 50, 297–313 (2010). http://www.preditec.com/mantenimiento-predictivo/

2.2.1 PRINCIPAL COMPONENTS ANALYSIS, PCA

Andrzej Mackiewicz y Waldemar Ratajczak; "Principal components analysis (PCA)", ScienceDirect; 1992.

D.R. Lewin; "Predictive maintenance using PCA"; ScienceDirect, 2000.

2.2.2 LINEAR VECTOR QUANTIFICATION, LVQ

Ali Yavari, Hossein Momeni, Fatemeh Goli and Mahmoud Ahmadi Chakoli; "Optimality Evaluation of Maintenance Strategy Using LVQ Neutral Network";Journal of Knowledge-Based Engineering and Innovation Vol. 1(1); 2015.

2.2.3 REDES Bayesianas

Mohamed Benzerga, "Introduction to Bayesian Networks and Predictive Maintenance – Part 1", https://expleogroup.medium.com/introduction-to-bayesian-networks-and-predictive-maintenance-part-1-831d22cad158, 2021.

2.2.4 Replicator Neuronal Network, RNN

Wari, "Predictive Maintenance Using Replicator Neural Network and Edge AI", Predictive Maintenance Using Replicator Neural Network and Edge AI | by Wari | Predictive Maintenance | Medium, 2020.

2.2.5 Dynamic Wavelet Neuronal Network, DWNN

P. Wang and G. Vachtsevanos, "Fault Prognosis Using Dynamic Wavelet Neural Networks", https://www.aaai.org/Papers/Symposia/Spring/1999/SS-99-04/SS99-04-018.pdf, 1999.

2.2.6 Self-Organized Feature Maps, SOM

John A. Bullinaria, "Introduction to Neutral Networks: Lecture 16" https://www.cs.bham.ac.uk/~jxb/NN/l16.pdf, 2004.

3.2 Programa principal e interfaz

https://shiny.rstudio.com/tutorial/