



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA



REPOSICIÓN AUTOMÁTICA EN EMPRESA DE RETAIL MEDIANTE ALGORITMOS
DE OPTIMIZACIÓN

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELÉCTRICO

ERICK FELIPE SALOMÓN PÉREZ FLORES

PROFESOR GUÍA:
DAVID VALENZUELA URRUTIA

PROFESOR CO-GUÍA:
FRANCISCO RIVERA S.

MIEMBROS DE LA COMISIÓN:
ALVARO SILVA M.

SANTIAGO DE CHILE
2021

Resumen

Resumen

dedicatoria corta.

Agradecimientos

Agradezco a todos.

Tabla de Contenido

1. Introducción	1
1.1. Problema	1
1.2. Situación actual	2
1.3. Objetivos	2
1.3.1. Objetivos Generales	2
1.3.2. Objetivos Específicos	2
1.4. Alcances	3
1.5. Estructura del trabajo	3
2. Marco teórico y estado del arte	4
2.1. Programación Lineal	4
2.1.1. Programación Lineal Entera	5
2.1.2. Algoritmo simplex	5
2.1.3. Algoritmo de puntos interiores	5
2.1.4. Branch and Bound	5
2.1.5. Planos cortantes	5
2.1.6. Formulación Big-M	5
2.2. Estado del arte: optimización en el retail	5
2.2.1. Solvers lineales	5
3. Metodología	6
3.1. Herramientas computacionales	6

3.2. Nomenclatura	7
3.3. Actividades	7
4. Desarrollo y resultados	10
4.1. Modelo	10
4.1.1. Función objetivo	10
4.1.2. Restricciones	11
4.2. Implementación del modelo	12
4.2.1. Implementación por ventanas	12
4.2.2. Implementación Naive	15
4.3. Pruebas sobre la implementación	17
4.3.1. Variación función objetivo	17
4.3.2. Pruebas de stress	18
4.4. Pruebas con entradas modificadas	20
4.4.1. Ruido en forecast de demanda	20
4.4.2. Distintas limitantes	22
5. Análisis	23
6. Conclusiones	24
6.1. Trabajo futuro	24
Bibliografía	25
Apéndice A. Parámetros	26

Índice de Tablas

3.1. Descripción de cada sigla utilizada en el trabajo.	7
3.2. Descripción y unidades de cada uno de los parámetros y variables utilizadas en el trabajo.	7
4.1. Rendimiento implementación con ventana 8 semanas vs caso Naive.	15
4.2. Rendimiento implementación con ventana 8 semanas $I = 2, J = 2$	17
4.3. Rendimiento implementación con ventana 8 semanas $I = 10, J = 10$	18
4.4. Complejidad se optimización.	18
4.5. Tiempos de ejecución, solver Gurobi.	19
4.6. Tiempos de ejecución, solver PuLP CBC.	19
4.7. Comparación de solvers.	19
4.8. Ruido en forecast de demanda $I = 2, J = 2, T = 8$	20
4.9. Ruido en forecast de demanda $I = 20, J = 20, T = 8$	22

Índice de Ilustraciones

4.1.	Ejemplo de reposición semanal durante 13 semanas, con ventana de temporal igual a 8.	13
4.2.	1	14
4.3.	1	14
4.4.	1	15
4.5.	1	16
4.6.	1	20
4.7.	1	21
4.8.	1	21
4.9.	1	22

Capítulo 1

Introducción

1.1. Problema

Para una empresa de retail es muy importante tener presente las preferencias e intereses de los compradores para así poder ofrecer los productos correctos, en el momento correcto, y en el lugar correcto. Dependiendo de los artículos que se tengan en los estantes de las tiendas, se tiene un mayor o menor ingreso, lo que hace relevante buscar estrategias para generar el mayor beneficio posible para la empresa, donde este beneficio está asociado al ingreso por venta de productos.

Para una empresa de retail con gran número de tiendas y SKU's a la venta, puede no aprovecharse todo el potencial de ventas cuando no se consideran todas las variables en la planificación. Una planificación inadecuada puede causar problemas de sobre-stock o quiebres de stock de algunos SKU, los que a su vez generan problemas de stock en otros productos o en otras tiendas, desaprovechando la oportunidad de satisfacer toda la demanda y generar el máximo beneficio. Una forma de tratar con este problema es utilizar la tecnología, que permite realizar tareas de alto computo en muy poco tiempo y de manera precisa. Utilizar algoritmos para la planificación de reposición de los distintos SKU a cada una de las tiendas permite atacar el problema de manera centralizada, evitando que algunas tiendas generen problemas de stock en otras. Al generar una reposición automática disminuyen las responsabilidades de los empleados, pasando de planificadores a supervisores, disminuyendo también las horas-persona necesarias para hacer la planificación. Por último, se destaca que este nuevo enfoque centralizado ya no se centra en cada tienda aislada, sino que se ve como un conjunto de éstas para generar un mayor beneficio para la empresa.


Se define el problema de manera matemática como una optimización, donde la función objetivo está relacionada con el beneficio económico de la empresa, y las restricciones tienen relación con las capacidades de almacenamiento y transporte de productos, además de restricciones lógicas. Como resumen de manera cualitativa:

Max. Beneficio económico para la empresa
s.a. Restricciones de Stock
Restricciones lógicas

Definir la forma en que se modelan los beneficios no es una tarea trivial, más bien es un desafío importante dentro del problema. Usualmente, para definir los beneficios se toman en consideración factores como *pricing*, predicciones de ventas, caracterización de artículos similares, sumado a un modelamiento de las preferencias de los compradores. También, se consideran los costos asociados a cada producto y el costo por tener por tiempos prolongados los productos almacenados en centro de distribución y tiendas.

Hablar de restricciones brevemente.

1.2. Situación actual

En esta sección se presenta el panorama actual de la empresa en esta materia.  ~~planificación está descentralizada donde cada una de las tiendas genera las cantidades de reposición de manera semanal.~~ Este trabajo está designado a cargo de personas, por lo que la generación de una buena planificación depende del talento de cada encargado. No se tiene una metodología generalizada para generar la planificación, cada tienda utiliza los recursos que se consideran adecuados.



1.3. Objetivos

1.3.1. Objetivos Generales

Desarrollar un algoritmo capaz de entregar las cantidades de artículos a reponer, en cada una de las tiendas del retail seleccionado, de manera centralizada y con un horizonte de semanas determinadas por el *forecast* de la demanda.

1.3.2. Objetivos Específicos

- Definir parámetros y datos a utilizar.
- Modelar función objetivo, definiendo claramente el beneficio.
- Definir costos, modelando correctamente el costo de oportunidad de tener indefinidamente un producto en bodega y en tienda.
- Definir restricciones del problema.

- Implementación de algoritmos de programación lineal (LP) para la optimización.
- Implementación de algoritmos de metaheurística para la optimización.
- Definir métricas de evaluación del rendimiento de cada algoritmo de optimización.
- Implementar herramientas que permitan analizar los resultados de optimización.
- Evaluar la validez de las soluciones encontradas, dada la realidad de la empresa.

1.4. Alcances

1.5. Estructura del trabajo

Capítulo 2

Marco teórico y estado del arte



2.1. Programación Lineal

La programación lineal es una de las grandes historias de éxito en la programación. Desde su formulación en los 1930s y 1940s y el desarrollo del algoritmo simplex a mediados de 1940, generaciones de trabajadores en economía, finanzas, e ingeniería han sido entrenados para formular y resolver problemas de programación lineal (LP). Incluso cuando las situaciones modeladas son no lineales, se utilizan para estas formulaciones lineales porque permiten la utilización de un software mas sofisticado, los algoritmos garantizan la convergencia, y porque incertezas en los modelos y datos comúnmente hacen impracticable construir un modelo no lineal más elaborado.

Las propiedades fundamentales de un problema de programación lineal son:

1. Un vector variables, el cual son obtenidos al resolver el problema de optimización.
2. Una función objetivo lineal.
3. Restricciones lineales, tanto igualdades como desigualdades

Se puede escribir de manera genérica de la siguiente forma:

$$\begin{array}{ll} \text{Encontrar vector} & \mathbf{x} \\ \text{que maximiza} & \mathbf{c}^T \mathbf{x} \\ \text{sujeto a} & \mathbf{Ax} \leq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{array}$$

donde \mathbf{c} y \mathbf{x} son vectores en \mathbb{R}^n , \mathbf{b} es un vector en \mathbb{R}^m , \mathbf{A} en una matriz de $m \times n$. Al conjunto de todos los puntos que satisfacen $\mathbf{Ax} \leq \mathbf{b}$, y $\mathbf{x} \geq 0$, se le conoce como la región factible, donde se busca la solución.

2.1.1. Programación Lineal Entera

2.1.2. Algoritmo simplex

2.1.3. Algoritmo de puntos interiores

2.1.4. Branch and Bound

2.1.5. Planos cortantes

2.1.6. Formulación Big-M

2.2. Estado del arte: optimización en el retail

El problema es ampliamente tratado en la bibliografía, lo que ayuda para tener referencias de modelamiento.

En la mayoría de los casos se trata como un problema de optimización lineal de enteros (ILP), en donde se utilizan *solvers* lineales y algoritmos de metaheurística por igual.

A pesar de tener buenas referencias, se presenta como desafío poder ajustar lo que ese ha hecho en trabajos anteriores a la realidad de la empresa y los recursos con los que se cuenta (humanos, computacionales y calidad de los datos).

2.2.1. Solvers lineales

Capítulo 3

Metodología

Esta sección presenta la metodología que se lleva a cabo en el desarrollo del trabajo. Cada **un** de los pasos definidos está estrechamente relacionado con los objetivos particulares del trabajo, y en consecuencia, con el objetivo general. A continuación se hace mención de las herramientas computacionales que se utilizan a lo largo del trabajo. Luego en la parte **M1** se presenta la nomenclatura que se utiliza en el siguiente capítulo, tanto en el desarrollo de modelos como en la muestra de los resultados de la implementación de los mismos. Finalmente se muestra un punteo de de cada una de las actividades consideradas para el desarrollo, detallando los pasos y herramientas a utilizar.

3.1. Herramientas computacionales

Python es un un lenguaje de programación interpretado y de alto nivel. Se caracteriza por ser un lenguaje donde prima la legibilidad de código y de fácil aprendizaje. Es un lenguaje de programación de código abierto que cuenta con librerías que permiten desde manipular valores hasta utilizar algoritmos de inteligencia artificial. En este trabajo se utiliza **Python 3.8.8** instalado mediante **Anaconda 3**. Para la manipulación de variables y arreglos se utiliza **Numpy 1.20.1**. Para el cálculo de tiempo de ejecución de código se utiliza la librería **time**. Para la realización de visualizaciones y gráficos se utiliza **matplotlib 3.3.4**.

Como se menciona en secciones anteriores, se utilizan solvers lineales en la implementación para realizar la optimización de la reposición. Se utiliza la librería **PuLP 2.5.1** para implementar los solvers COIN-OR CBC, PuLP CBC y Guroby 9.5.0, estos 2 últimos utilizando una licencia académica.

Las simulaciones son ejecutadas en 2 tipos de computadores con procesadores: Ryzen 5 4650 PRO con **graficos ibntegrados** vega 7, 16GB RAM y sistema operativo Windows 10 (llamese de ahora en adelante R5) y un procesador Intel Core i5, 8GB RAM y sistema operativo Ubuntu 20.04 (llamese de ahora en adelante i5).

3.2. Nomenclatura

Tabla 3.1: Descripción de cada sigla utilizada en el trabajo.

Sigla	Significado
SKU	Stock keeping unit
LP	Programación lineal
ILP	Programación lineal de enteros
R5	Ryzen 5 4650 PRO, 16GB RAM y sistema operativo Win10
i5	Intel Core i5, 8GB RAM y sistema operativo Ubuntu

Tabla 3.2: Descripción y unidades de cada uno de los parámetros y variables utilizadas en el trabajo. Describe también los índices de dependencia de cada una: i (depende del SKU), j (depende de la tienda) y t (depende de la semana).

	Descripción	Índices	Unidad
R	Reposición	i, j, t	
F	Forecast de demanda	i, j, t	
P	Precio	i, j, t	CL\$
\tilde{Q}	Demanda ajustada	i, j, t	
C	Costo	i, j, t	CL\$
δ	Parámetro binario lógico	i, j, t	
S_{CD}	Stock en centro de distribución	i, t	
F_{vol}	Volumen de cada sku	i, j, t	cm ³
T	Tamaño de ventana		semanas
\tilde{t}	$T - t$ tiempo modificado		semanas
I	Cantidad de sku		
J	Cantidad de semanas		
Inv	Inventario	i, j, t	
B	Capacidad en tienda	j, t	cm ³
Me	Mínimo de exhibición	i, j, t	
Tr	Límite máximo de transporte	t	cm ³

3.3. Actividades

La metodología se divide en 3 partes que se pueden nombrar como: modelamiento, implementación y evaluación. A continuación se describe el enfoque que tiene cada uno y sus principales actividades. El detalle de las actividades de las actividades se encuentra de manera más amplia en la sección 4.

Modelamiento contempla la definición del modelo de optimización. Se define que es un modelo lineal y por ende se busca una función objetivo lineal y restricciones lineales. Es importante también definir de buena manera las variables y parámetros del problema, para obtener la solución deseada, unas mala elección de restricciones, o variables de decisión,

puede llevar a un algoritmo poco optimizado, significando un mayor nivel de procesamiento y mayores tiempos de ejecución.



Implementación hace alusión a los algoritmos a utilizar para resolver, de manera computacional, el problema modelado. En este caso los algoritmos son implementados en python utilizando solvers lineales para la optimización. También dentro de la implementación están las herramientas para visualizar los resultados y obtener métricas que permitan validar y evaluar, de manera adecuada, el modelo implementado.

Evaluación corresponde a la parte final del proceso en donde da una valoración a cada implementación del modelo. La metodología para lograr lo anterior es probar con varias configuraciones de parámetros, variables, implementaciones, solvers, entre otros, obteniendo distintos valores de las métricas. Mediante esto, se puede concluir acerca del desempeño del trabajo realizado.

Las actividades de las cuales se entra en más detalle en la sección 4 son:

- **Función objetivo:** Se define la función objetivo con los términos que la componen. Cada uno de estos debe representar un sentido intuitivo, de la solución que se busca. Para moldear la solución y descartar casos indeseados, se utiliza la función objetivo para empujar el óptimo a un calendario de reposición más realista.
- **Restricciones:** Se busca definir restricciones que sean lo más generales posibles, con el fin de que sean rígidas. Para realizar la evaluación, se utilizan distintas configuraciones de la función objetivo, sin embargo, las restricciones se mantienen fijas durante todo el trabajo. Por lo tanto, las restricciones están asociadas principalmente a dar sentido al problema, ~~pero no se innova mucho~~, utilizando restricciones estándar como las que se encuentran en [?].
- **Variables:** Se definen las variables del problema, entre las cuales debe estar la reposición semanal.
- **Parámetros:** Corresponde a las entradas que recibe el modelo. Se diferencian en dos tipos: el primero corresponde a entradas modificables por el usuario, y los segundos son parámetros que están fijos y no los controlan los usuarios. En esta sección se detallan los parámetros que son utilizados en las simulaciones implementadas.
- **Implementación por ventanas:** Para realizar la optimización toman en cuenta una cantidad determinada de semanas que el modelo conoce por adelantado, con el fin de anticiparse a futuros peak de demanda, con esto se puede distribuir de mejor manera el stock en tiendas, de manera que no hayan quiebres de stock en las semanas de mayor demanda.
- **Implementación Dummy:** Se implementa un algoritmo que no conoce el la demanda futura, lo que puede provocar quiebres de stock.
- **Implementación función objetivo:** Se implementa el mismo modelo por ventanas mencionado anteriormente pero esta vez con cada una de las combinaciones de función objetivo posibles. Esto con el objetivo de evaluar el tiempo de ejecución, cantidad de ventas y cantidad de ganancias dada las distintas configuraciones.

- **Pruebas de stress:** Se implementa el mismo modelo por ventanas mencionado anteriormente, variando la cantidad de SKU y tiendas en el calendario de reposición, para obtener el desempeño en tiempo de ejecución del algoritmo. Adicionalmente, esto se realiza para distintos solvers, para comparar el desempeño de estos.
- **Pruebas de ruido en forecast de demanda:** Se agrega ruido al forecast de demanda para simular el error al generar éste, presente en cualquier forecast. Se evalúa mediante cantidad de quiebres que se obtienen en función del nivel de ruido que se introduce.
- **Prueba de implementación en distintas situaciones:** Dependiendo de los parámetros utilizados, el calendario de reposición se comporta de manera distinta, por lo que se realizan combinaciones de parámetros que entregan situaciones intuitivas e ilustrativas para sacar conclusiones de la implementación en estas situaciones.

Capítulo 4

Desarrollo y resultados

Para hacer alusión a los objetivos del trabajo, se busca realizar simulaciones, implementando algoritmos de optimización, para conseguir un calendario de reposición de una cantidad determinada de SKUs a una cantidad determinada de **tiendas. en este** sentido, se presenta primero el modelo realizado para optimizar, que tiene como salida el calendario de reposición semanal. Posteriormente se implementa **el python** distintos solvers lineales que permiten realizar la optimización del modelo señalado anteriormente. Por último se implementan una serie de algoritmos que permiten variar los parámetros para evaluar el desempeño del modelo desarrollado ~~y su implementación.~~

4.1. Modelo

Se presenta en esta sección el modelo de optimización que se desarrolla en este trabajo, esto incluye las variables y parámetros involucradas, como así también los términos de la función objetivo y el significado de las restricciones. Toda la notación utilizada se encuentra de manera detallada en el **Apéndice A** y **se** manera resumida en la Tabla 3.2.

4.1.1. Función objetivo

Se presenta en la ecuación 4.1 la función objetivo que se utiliza para modelar el problema de optimización.

$$\text{Max.} \sum_{i,j,t} \underbrace{\tilde{Q}(P-C) \cdot \tilde{t}^2}_{Z_1} + \underbrace{\tilde{Q} \cdot \tilde{t}^2}_{Z_2} + \underbrace{\delta \cdot F \cdot \tilde{t}^2}_{Z_3} - \underbrace{S_{CD} \cdot F_{vol} \cdot t^2}_{Z_4} \quad (4.1)$$

Se pueden identificar 4 términos en la función objetivo **4.1**, **Z_1** corresponde al beneficio económico que se obtienen por ganancias de la venta de productos, en este caso se tiene que la ganancia de venta de un producto es el precio de venta P menos el costo de éste C por la demanda \tilde{Q} . Se busca que este término haga que la solución prefiera la venta

y reposición de productos que generan mejores ganancias, así siempre se tendrá stock en tiendas de estos productos. Sólo implementar este término puede producir que productos que generan menores ganancias sean desfavorecidos en la reposición, generando que productos específicos no se vendan.

El término Z_2 da prioridad a vender la mayor cantidad de productos lo antes posible, sin importar la ganancia que generen. Sólo tener este término puede generar efectos negativos en la solución para la reposición, en lugar de generar una reposición homogénea, prioriza productos con demandas altas sin importar la ganancia que estos generan, dejando con poco stock en tiendas los productos de menor demanda.

El término Z_3 es mayor que cero sólo cuando se cumple la demanda, cuando se genera un quiebre de stock este término tiene valor cero. Su inclusión en el modelo aporta, de manera explícita, una preferencia por las soluciones que entregan la menor cantidad de quiebres de stock posibles, situaciones indeseadas por los retailers.

Por último, el término Z_4 es un término que cumple una función de costo, la razón de tener un signo negativo antes de este término. Este costo está asociado a mantener los productos por tiempos prolongados en el centro de distribución, provocando que la solución encontrada busque tener tiendas llenas, o a la cota seleccionada mediante el parámetro B .

Cabe destacar que el factor \tilde{t} presente en los 3 primeros términos tiene como objetivo que se prioricen cumplir la demanda los periodos semanales más próximos.



4.1.2. Restricciones



Las restricciones implementadas están directamente relacionadas con lo que se encuentra a lo largo de la bibliografía, como también a la realidad de la empresa y sus necesidades. La restricción 4.2 y 4.3 y 4.8 no superar límites establecidos para que el problema tenga sentido y se ajuste a las demandas del retailer, la primera pretende no superar la capacidad de almacenaje por producto que se tiene en las tiendas, la segunda no superar el stock de cada SKU que hay en el centro de distribución y la tercera no superar la capacidad de transporte semana. La restricción 4.4 indica que las cantidades no pueden ser negativas. Las restricciones 4.5 y 4.6 corresponden a restricciones de continuidad para que las variables Inv y S_{CD} del modelo tengan sentido. La restricción 4.7 guarda relación con necesidades del retailer mantener stock en tiendas para satisfacer no sólo la demanda, sino que también un mínimo se exhibición. La última restricción 4.9 tiene una connotación lógica e indica que la demanda solo se puede suplir cuando el stock en la tienda es suficiente.



$$\sum_i (Inv_{i,j,(t-1)} + R_{i,j,t}) \leq B_{j,t} \quad \forall j \in [[1, J]], \forall t \in [[1, T]] \quad (4.2)$$

$$\sum_j R_{i,j,t} \leq S_{CD \ i,t} \quad \forall i \in [[1, I]], \forall t \in [[1, T]] \quad (4.3)$$

$$R_{i,j,t} \geq 0 \quad \forall i \in [[1, I]], \forall j \in [[1, J]], \forall t \in [[1, T]] \quad (4.4)$$

$$Inv_{i,j,t-1} + R_{i,j,t} - \tilde{Q}_{i,j,t} - Inv_{i,j,t} = 0 \quad \forall i \in [[1, I]], \forall j \in [[1, J]], \forall t \in [[1, T]] \quad (4.5)$$

$$S_{CD \ i,t} + S_{CD \ i,(t-1)} + \sum_j R_{i,j,t} = 0 \quad \forall i \in [[1, I]], \forall t \in [[1, T]] \quad (4.6)$$

$$Me_{i,j,t} \leq I_{i,j,t-1} \quad \forall i \in [[1, I]], \forall j \in [[1, J]], \forall t \in [[1, T]] \quad (4.7)$$

$$\sum_i \sum_j R_{i,j,t} \cdot F_{vol \ i} \leq Tr_t \quad \forall t \in [[1, T]] \quad (4.8)$$

$$\min (Inv_{i,j,t-1} + R_{i,j,t}, F_{i,j,t}) = \tilde{Q}_{i,j,t} \quad \forall i \in [[1, I]], \forall j \in [[1, J]], \forall t \in [[1, T]] \quad (4.9)$$

4.2. Implementación del modelo

4.2.1. Implementación por ventanas

Para evitar **quiebres de stock** se implementa un modelo que utilice el forecast de la demanda futura, lo que permite equipar las tiendas con las cantidades que aseguren un cumplimiento de la **demanda**. Se implementa un algoritmo que realice la optimización con $T = 8$, lo que permite anticiparse a la demanda con **2 meses de anticipación aproximadamente**. Luego de realizar la optimización del modelo y obtener un calendario de reposición para esas 8 **semanas**, sólo se fija la primera y las otras 7 semanas se descartan. De manera ilustrativa se tiene la figura 4.1, que contiene la información de la reposición (R), inventario (Inv), forecast de demanda (F) y la **demanda cumplida** (\tilde{Q}) para cada semana. La figura indica que se ha **utilizado el modelo 13 veces (13 semanas)**, donde también se presenta la última optimización que va de la semana 13 a la 20 (ventana de 8 semanas) y de donde se fija sólo la semana 13 y las variables desde la semana 14 a la 20 se descartan, esperando fijarse en futuras iteraciones. Se normalizan las curvas, donde todos los valores en el gráfico están divididos por el máximo **den él**.

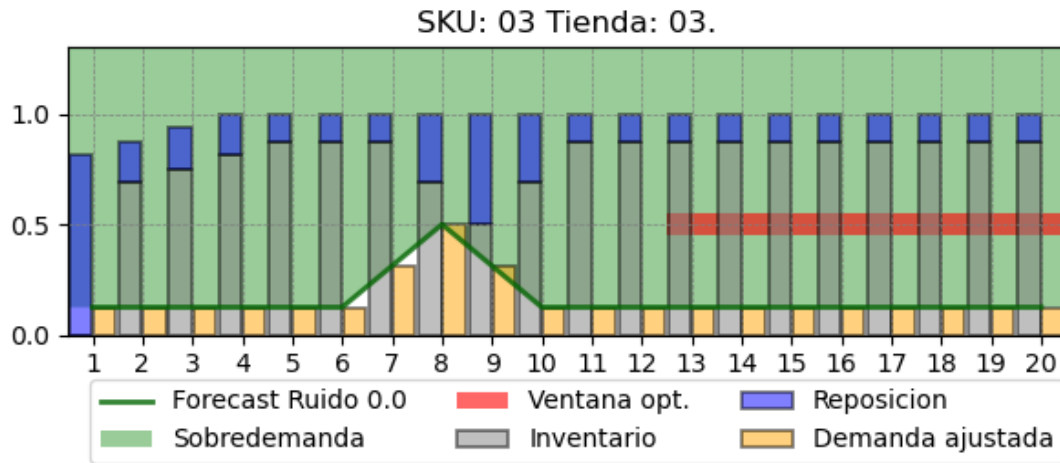


Figura 4.1: Resultados para SKU 03 en tienda 03.

Para todas las simulaciones que se presentan en el trabajo se tienen una curva de demanda con un peak de ventas en la semana 8. Entre las semanas 1-6 y 10-20, se considera una demanda constante y las semanas 7 y 9 hacen una transición lineal al peak.

La implementación de esta sección, y la siguiente 4.2.2, se realizan en R5 (ver Tabla3.1) teniendo como resultado las figuras 4.2 y 4.3. La implementación cuenta con 2 SKU y 2 tiendas, el resultado de la optimización, y las variables de cada tienda/SKU se encuentran en los 4 gráficos de la figura 4.2, en donde se puede apreciar que se hay quiebres de stock en 3 de los 4. Para el caso del SKU 01 - Tienda 01 hay un quiebre en la semana 8, cuando aumenta la demanda, y 2 quiebres en las semanas 19 y 20 generados por falta de stock general del SKU 01, donde ya no queda en el CD. En el caso del SKU 01 en la Tienda 02, se generan quiebres en las semanas 7, 8 y 9, semanas en que aumenta la demanda. Los dos gráficos superiores de la figura 4.3 corresponden a la ocupación de la tienda correspondiente, el valor 1 corresponde a la tienda a capacidad completa. El stock en el centro de distribución se ve en la misma figura, abajo a la izquierda, donde se ve que el stock de ambos SKU disminuyen y el SKU 01 se agota en la semana 20. Por último, abajo a la derecha, se aprecia la reposición de cada SKU a cada tienda de manera semanal, en ninguna semana se alcanza el máximo a repartir por tienda.



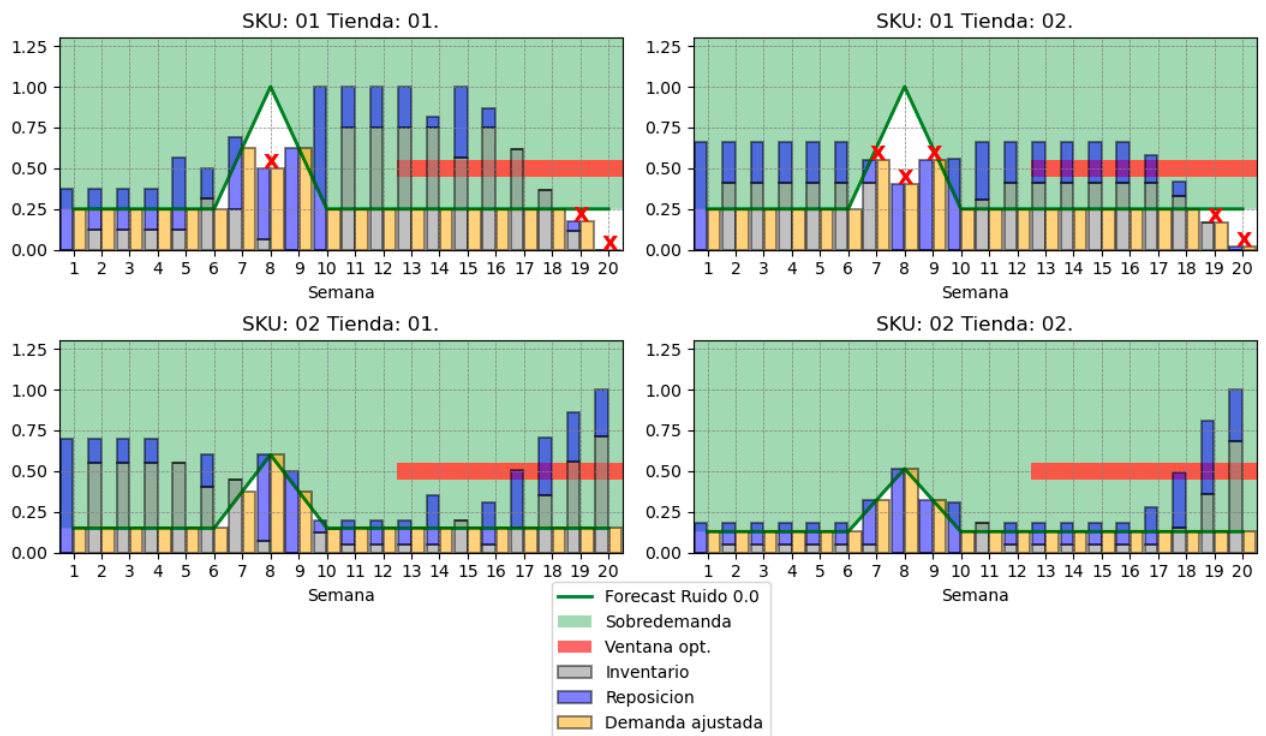


Figura 4.2: Resultados

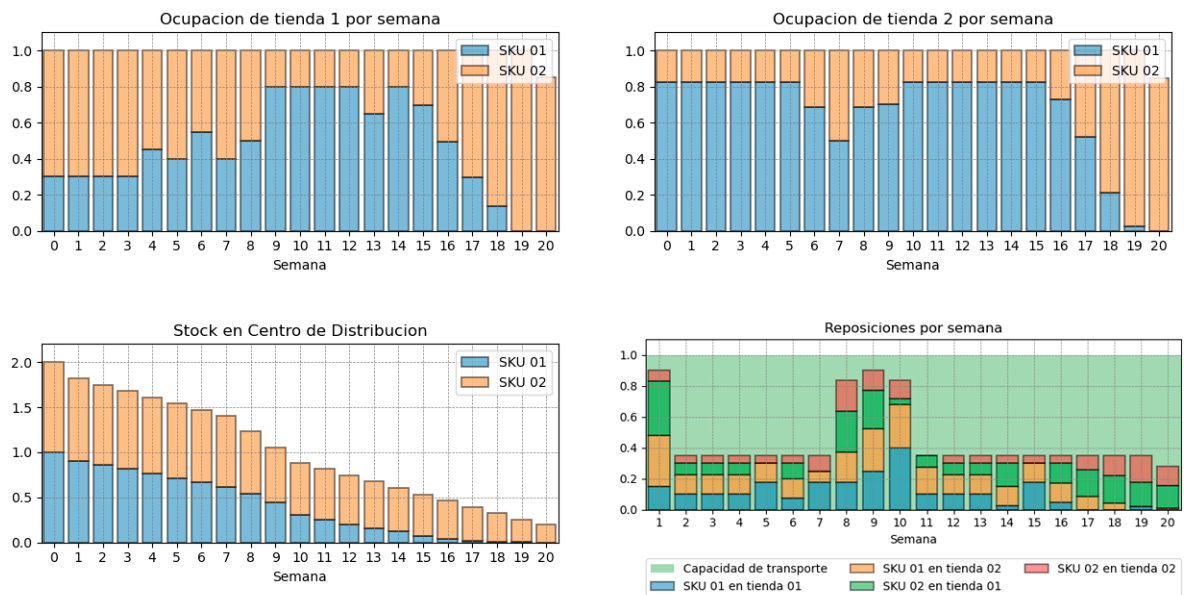


Figura 4.3: Resultados

4.2.2. Implementación Naive

Se utiliza la misma implementación que en la sección, variando sólo el tamaño de la ventana, que en este caso es 1. Se llama a esta implementación *Naive* debido a que en cada iteración semanal no ve la demanda futura, por lo que ingenuamente puede asignar reposiciones que traen un mayor beneficio en la *inmediatez*, sin embargo pueden causar quiebres de stock a futuro a causa de una mala distribución del stock, de cada SKU, dentro de las tiendas. Se presenta a continuación, en las figuras 4.3 y 4.3, la implementación descrita anteriormente.

Tabla 4.1: Valores considerando las demanas desde la 1 a la 13 de las implementaciomen mostradas en

T	Ganancias [MCL\$]	Unidades	Quiebres	Tiempo
1	41.9	11249	9	0.08
8	42.4	11450	4	0.32

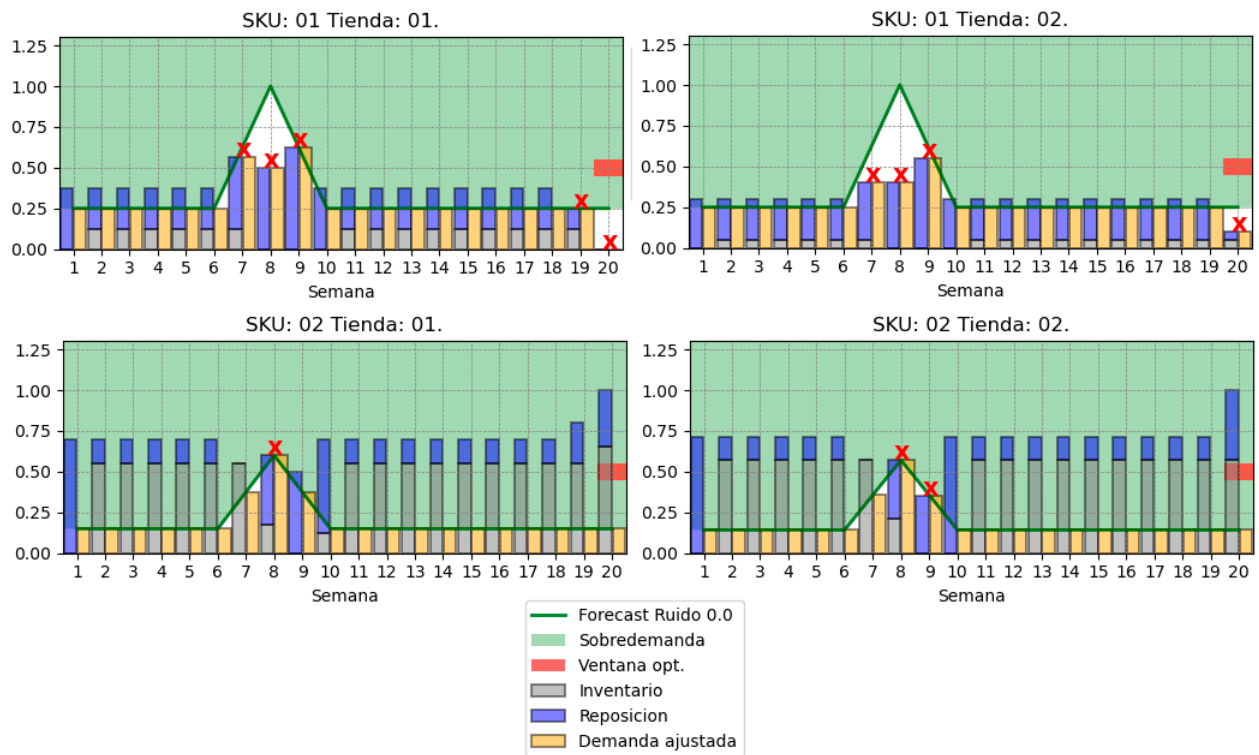


Figura 4.4: Resultados de la reposición de cada tienda y SKU, implementación Naive.

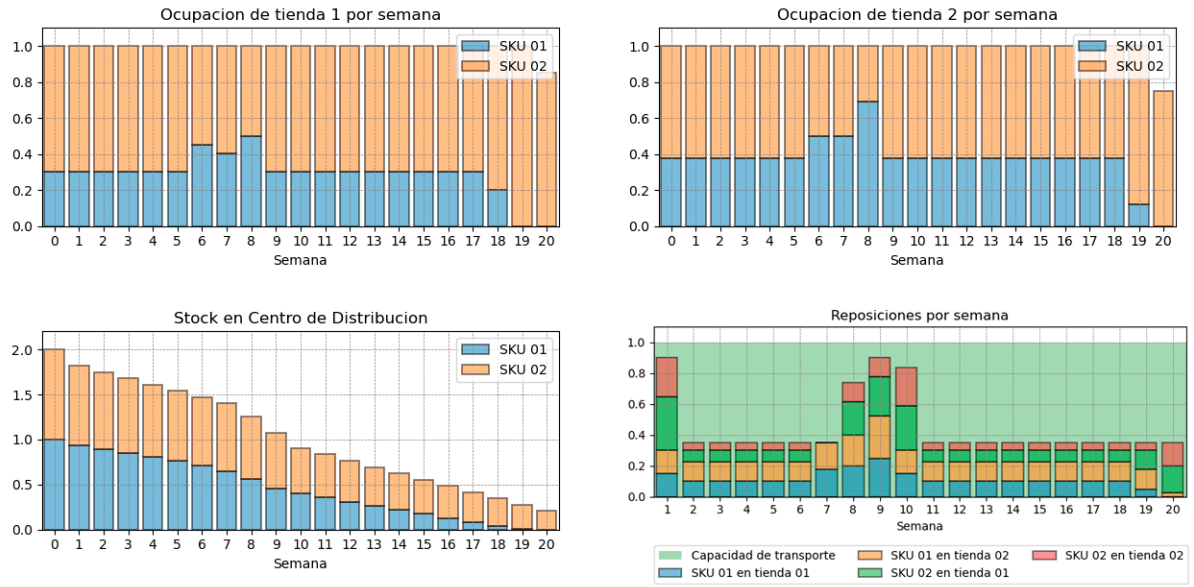


Figura 4.5: Resultados complementarios de la reposición de cada tienda y SKU, implementación Naive.

Si bien los valores en los gráficos se presentan normalizados, la Tabla 4.1 muestra los valores obtenidos en la implementación de esta sección y la anterior (Sección 4.2.1). Se puede apreciar que al agregar una ventana temporal más grande se puede anticipar a demandas futuras y reponer en consecuencia, lo que se traduce en mayor cantidad de unidades vendidas, mayores ingresos y menos quiebres de stock. Sin embargo, tener una ventana temporal de 8 semanas genera un algoritmo más complejo, demorando 3 veces más para 2 SKUs y 2 tiendas.

4.3. Pruebas sobre la implementación

4.3.1. Variación función objetivo

En esta sección se ven los resultados al probar con cada una de las combinaciones de términos el función objetivo mostrada en 4.1. Los resultados que se muestran en la Tabla 4.2, tienen la misma combinación de parámetros que en la Sección 4.2.1, sólo varía la función objetivo. La implementación se realiza con el solver Gurobi. Se destacan algunos valores de la Tabla 4.2: cuando no de tiene el término Z_1 (término que incluye precios y costos) se ven disminuidas las ganancias económicas en la mayoría de los casos. También cuando se incluye Z_2 sin el término Z_1 , en general, se mantiene la cantidad de unidades vendidas dado el forecast.

Tabla 4.2: Valores considerando las demanas desde la 1 a la 13 de las implementaciomen mostradas en

F.O.	Ganancias [MCL\$]	Unidades	Quiebres	Tiempo [s]
Z_1	42.38	11450	4	0.24
Z_2	38.92	11450	5	0.24
Z_3	37.67	10610	4	0.25
Z_4	42.07	11279	4	0.29
$Z_1 + Z_2$	42.39	11450	4	0.24
$Z_1 + Z_3$	42.39	11450	4	0.24
$Z_1 - Z_4$	42.39	11450	4	0.29
$Z_2 + Z_3$	39.29	11220	4	0.29
$Z_2 - Z_4$	42.07	11280	4	0.28
$Z_3 - Z_4$	39.29	11219	4	0.30
$Z_1 + Z_2 + Z_3$	42.39	11450	4	0.24
$Z_1 + Z_2 - Z_4$	42.39	11450	4	0.23
$Z_1 + Z_3 - Z_4$	42.38	11448	4	0.25
$Z_2 + Z_3 - Z_4$	39.29	11220	4	0.32
$Z_1 + Z_2 + Z_3 - Z_4$	42.39	11450	4	0.32
Promedio	41.18	11325	4.1	0.27
Mínimo	37.67	10610	4	0.23
Máximo	42.39	14450	5	0.32
Desviación Estándar	1.66	214	0.25	0.03

En la Tabla 4.3 se tienen resultados par aun mayor número de SKUs y tiendas, con el objetivo de hacer más notorias las diferencias y ver patrones más marcados en el desempeño de cada función objetivo. Se destaca que la desviación estándar de las ganancias, y las unidades, es menor al 1 %. La desviación estándar del número de quiebres es del 24 %. El tiempo de ejecución es la métrica de mayor variabilidad, donde el máximo supera en más de 3 veces al promedio.

Tabla 4.3: Valores considerando las demanas desde la 1 a la 13 de las implementaciones mostradas en

F.O.	Ganancias [MMCL\$]	Unidades	Quiebres	Tiempo [s]
Z_1	1252	319709	18	5.33
Z_2	1245	318844	24	4.77
Z_3	1239	317549	24	56.88
Z_4	1239	318447	25	7.11
$Z_1 + Z_2$	1256	320666	18	3.87
$Z_1 + Z_3$	1252	319385	18	13.82
$Z_1 - Z_4$	1258	321358	15	6.13
$Z_2 + Z_3$	1228	314927	28	54.26
$Z_2 - Z_4$	1251	320554	26	5.72
$Z_3 - Z_4$	1226	314811	29	13.71
$Z_1 + Z_2 + Z_3$	1254	320158	17	12.70
$Z_1 + Z_2 - Z_4$	1250	322168	14	7.23
$Z_1 + Z_3 - Z_4$	1250	318768	20	17.87
$Z_2 + Z_3 - Z_4$	1240	318335	22	10.16
$Z_1 + Z_2 + Z_3 - Z_4$	1254	320192	17	11.49
Promedio	1246	319058	21	15.40
Mínimo	1226	314811	14	3.87
Máximo	1258	322168	29	56.88
Desviación Estándar	9	2020	5	16.23

4.3.2. Pruebas de stress

Es importante conocer la complejidad de un algoritmo, que se traduce en mayor uso de memoria y tiempo de computo. En la Tabla 4.4 se presenta la cantidad de variables de decisión y cantidad de constantes que participan en la optimización. También se señalan la cantidad de restricciones del problema, que dependen de I , J y T .

En las Tablas 4.5 y 4.6 los tiempos de ejecución de una iteración de optimización para la reposición, con una ventana temporal de 8 semanas, variando la cantidad de SKUs y tiendas, utilizando dos solvers distintos: Gurobi y PuLP CBC, respectivamente.

Tabla 4.4: I cantidad de SKUs y J cantidad de tiendas, con ventana temporal de T

Variables	Constantes	Restricciones
$4 IJT + IT$		$7 IJT + 2 IT + JT + T$



Tabla 4.5: Tiempos de ejecución, solver Gurobi, $T = 8$

I	2	8	15	20	25	30	40	50
2	0.01	0.07	0.10	0.10	0.13	0.16	0.22	0.29
6	0.05	0.15	0.27	0.38	0.45	0.68	0.88	1.14
10	0.07	0.26	0.48	0.66	0.90	1.05	1.61	2.29
20	0.13	0.52	1.14	1.64	2.15	2.79	4.32	6.01
50	0.38	1.59	3.90	8.66	12.19	13.84	38.37	27.12
100	0.76	3.37	9.89	27.90	58.01	37.63	114.90	189.45
200	1.59	20.19	31.56	133.42	196.84	>200	>200	>200
500	8.37	54.82	153.99	>200	>200	>200	>200	>200

Tabla 4.6: Tiempos de ejecución, solver PuLP CBC, $T = 8$

I	2	8	15	20	25	30	40	50
2	0.06	0.26	0.28	0.41	0.47	0.57	0.87	1.25
6	0.12	0.63	1.28	10.97	2.30	5.3	3.26	>200
10	0.15	0.76	2.27	2.53	3.97	5.82	7.55	>200
20	0.42	2.15	5.31	7.19	7.87	10.03	21.11	40.03
50	1.11	6.41	24.02	31.83	>200	67.38	149.28	>200
100	2.35	21.63	78.95	150.52	>200	>200	>200	>200
200	3.02	124.08	>200	>200	>200	>200	>200	>200
500	10.81	>200	>200	>200	>200	>200	>200	>200

Tabla 4.7: Comparación de solvers, $I = 10$, $T = 8$

I	2	8	15	20	25	30	40	50
PuLP CBC	0.15	0.76	2.27	2.53	3.97	5.82	7.55	>200
COIN-OR								
Gurobi	0.07	0.26	0.48	0.66	0.90	1.05	1.61	2.29

4.4. Pruebas con entradas modificadas

4.4.1. Ruido en forecast de demanda


El modelo debe ser robusto a variaciones en los parámetros, ~~por lo que se hace necesario evaluar el modelo, y su implementación, induciendo error en sus parámetros.~~ El único parámetro que tiene un error considerable es el forecast de la demanda  para poder evaluar la robustez se procede optimizando de la misma manera que en la Sección 4.2.1, **donde** semana a semana **se corrigen las \tilde{Q}** añadiendo ruido al forecast de la demanda F , **lo que** permite evaluar métricas como número de quiebres, ganancias y cantidad de unidades, cuantificando la diferencia de tener error en el forecast. El ruido se modela como una distribución normal de media $\mu = F_{i,j,t}$ y desviación estándar $\tilde{\sigma} = \mu * \sigma$. Las pruebas que se realizan en esta sección, corresponden a implementar el algoritmo variando el valor de σ , cabe destacar que con $\sigma = 0$ se recupera el caso original sin ruido presentado en la Sección 4.2.1.

Tabla 4.8: **Desempeño con ruido en forecast** de demanda I = 2, J = 2, T = 8

σ	Ganancias [MCL\$]	Unidades	Quiebres
0.0	42.38	11450	4
0.2	41.19	11370	6
0.6	38.08	10931	9

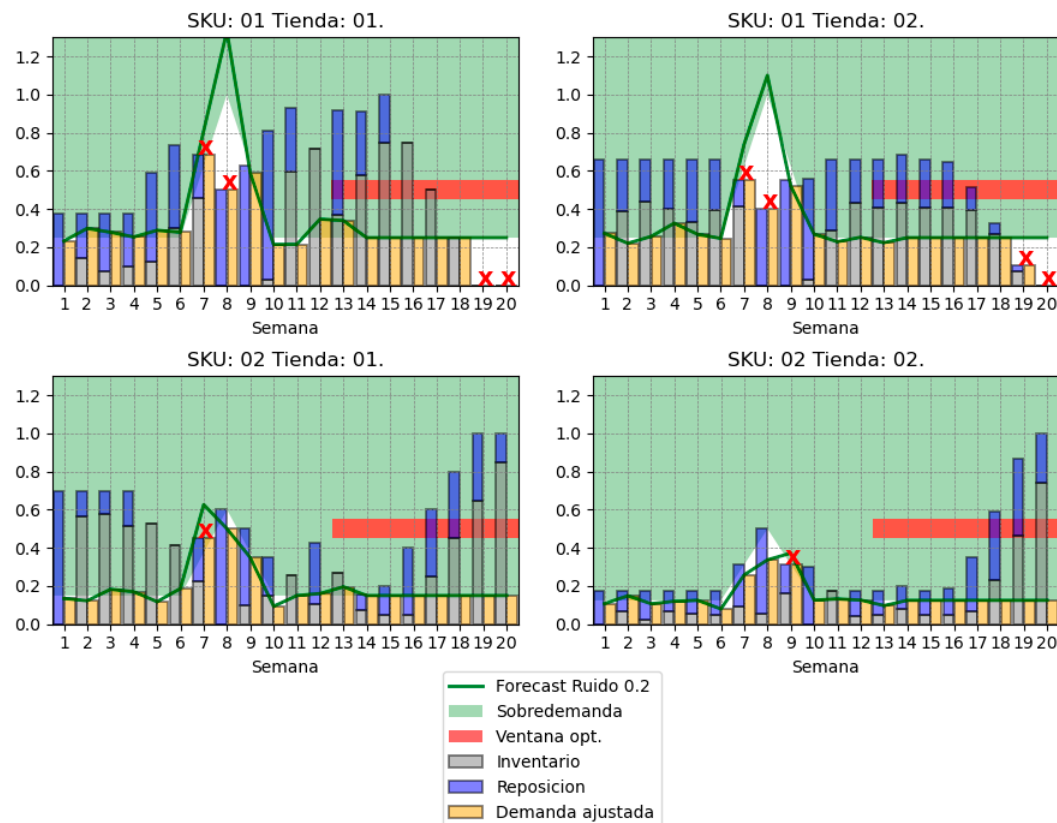


Figura 4.6: Resultados ruido 0.2

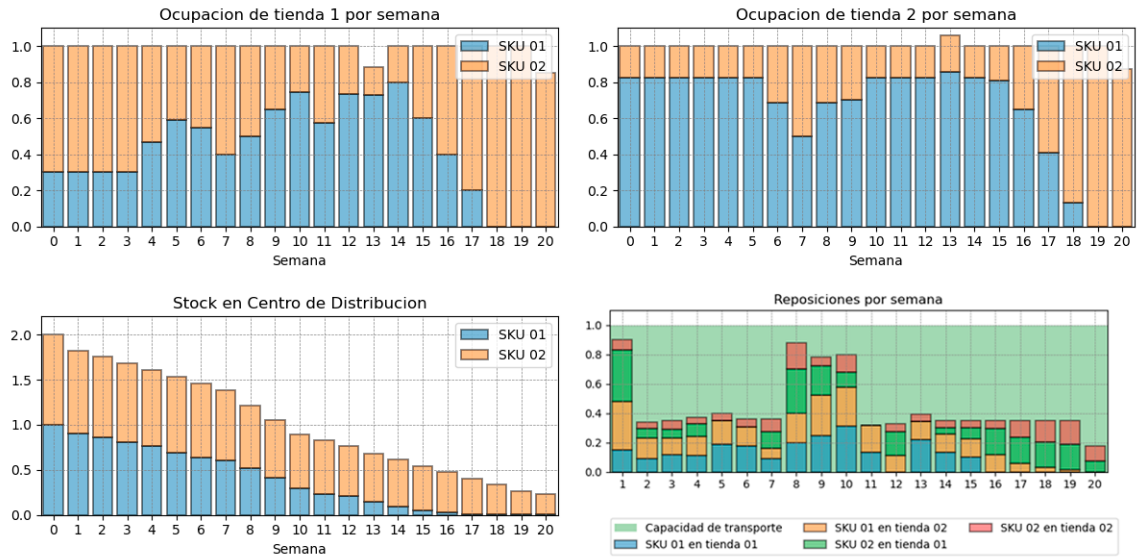


Figura 4.7: Resultados complementarios ruido 0.2

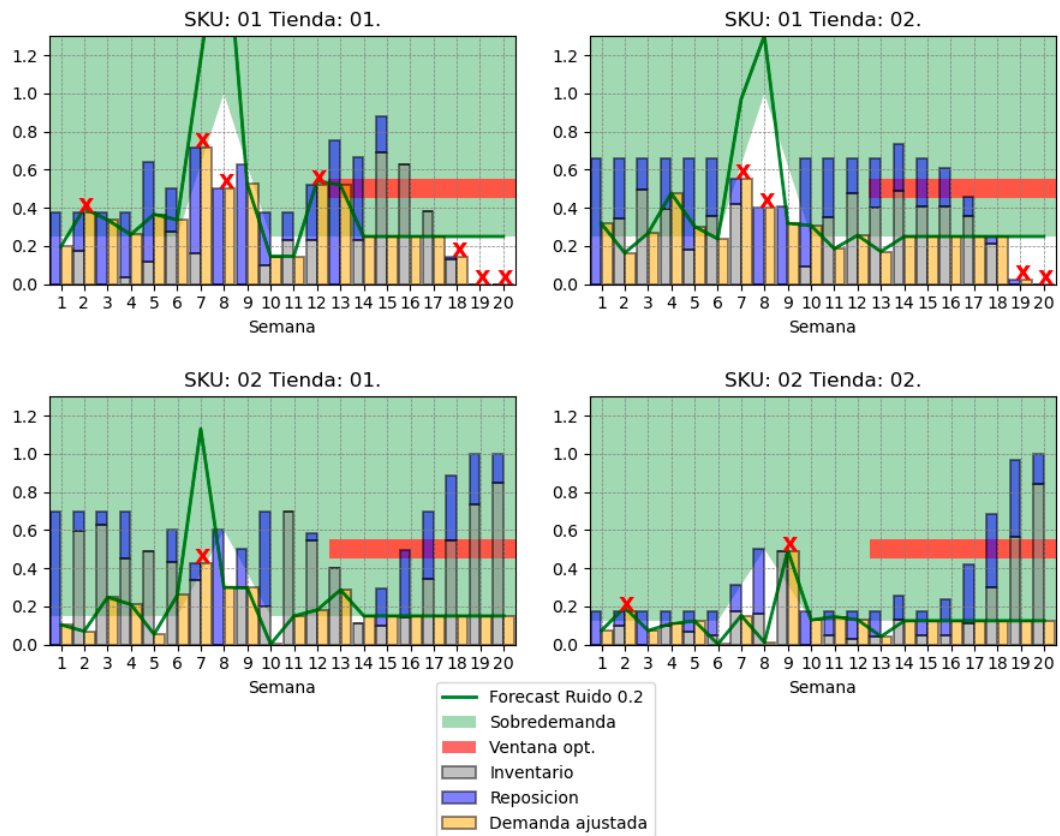


Figura 4.8: Resultados ruido 0.6

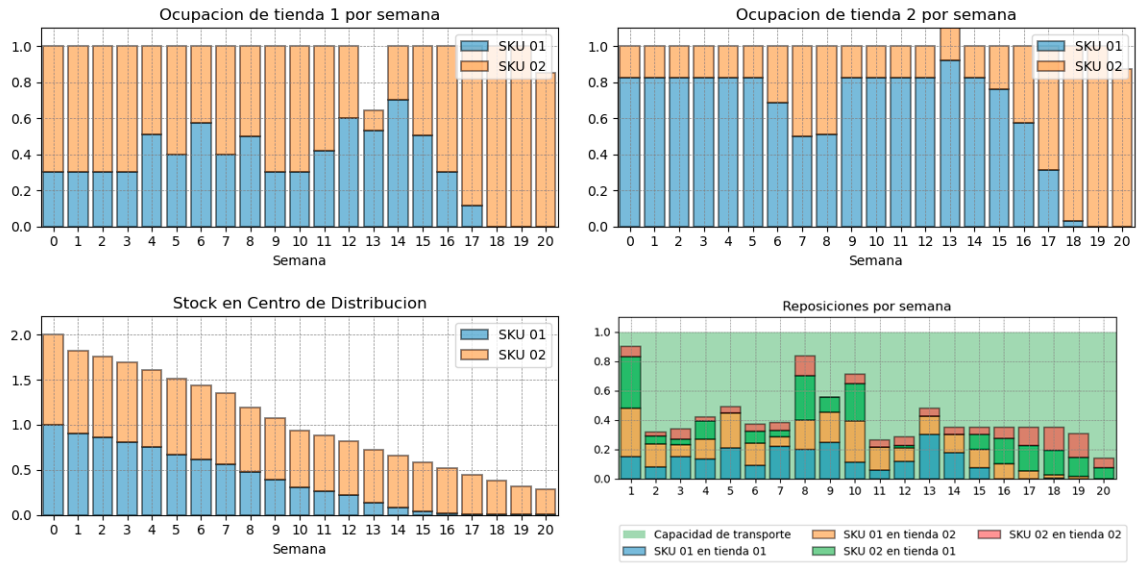


Figura 4.9: Resultados complementarios ruido 0.6

Tabla 4.9: Desempeño con ruido en forecast de demanda $I = 20$, $J = 20$, $T = 8$

σ	Ganancias [BCL\$]	M Unidades	Quiebres
0.0	5.10	1.29	67
0.1			
0.2			
0.3			
0.4			
0.5			
0.6			
0.7			
0.8			
0.9			
1.0			

4.4.2. Distintas limitantes

Capítulo 5

Análisis

Restricciones lógicas en el modelo funcionan, se ve en inventario parara el otro día, demanda cumplida, SCD disminuye lo que se reparte.

Capítulo 6

Conclusiones

6.1. Trabajo futuro

Bibliografía

- [1] Ioannis Karatzas. and Steven E. Shreve. *Brownian Motion and Stochastic Calculus*. Springer, Berlin, 2nd edition, 2000.
- [2] Philip Protter. *Stochastic Integration and Differential Equations*. Springer, 1990.
- [3] Daniel Revuz and Marc Yor. *Continuous martingales and Brownian motion*. Number 293 in Grundlehren der mathematischen Wissenschaften. Springer, Berlin [u.a.], 3. ed edition, 1999.

Apéndice A

Parámetros

Se detalla en este anexo cada uno de los parámetros utilizados en el modelo, se puede ver un resumen en la tabla 3.2.