

## ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

**GRADO EN INGENIERÍA INFORMÁTICA**

**Curso Académico 2023/2024**

**Trabajo Fin de Grado**

## DESARROLLO Y OPTIMIZACIÓN DE UN EXPERT ADVISOR BASADO EN EL INDICADOR DE ICHIMOKU Y EL RSI

**Autor**: Sergio Bolaños Díaz

**Tutor**: Carlos Grima Izquierdo

**RESUMEN**

Se ha desarrollado un Expert Advisor con el objetivo de obtener rentabilidad en los mercados financieros. Para lograr esto, se han empleado los indicadores de Ichimoku y RSI. Estos dos indicadores se adaptan a cualquier mercado, pero funcionan especialmente bien en mercados tendenciales. Por tanto, el objeto de estudio serán los mercados que cumplen estas condiciones.

Para operar en diferentes mercados se deben tener en cuenta los valores de los parámetros empleados. En función de ellos, se entrará o saldrá del sistema. Para entrar en una posición larga, el precio debe superar el límite superior de la nube de Ichimoku y el valor del RSI debe situarse por debajo de la zona de sobrecompra. Para salir de la posición larga, el precio debe cruzar el límite inferior de la nube de Ichimoku y el RSI debe estar por encima de la zona de sobreventa.

Las condiciones de entrada y salida para las posiciones cortas, son las mismas que para las largas pero invertidas. Es decir, para entrar en una posición corta, el precio debe cruzar hacia abajo la nube y el RSI debe estar por encima de la zona de sobreventa. Para salir de la posición corta, el precio debe cruzar hacia arriba el límite superior de la nube y el RSI debe estar por debajo de la zona de sobrecompra.

Es importante destacar que el sistema está programado para operar en la apertura de velas. Esto significa que se toman los valores de los indicadores a partir de la vela anterior. Si no nos encontramos en la apertura de la vela, no se opera.

Para el desarrollo del sistema se ha utilizado el entorno de programación MetaEditor4 y el correspondiente lenguaje de programación MQL4. Para la optimización y backtesting, se ha empleado la plataforma de MetaTrader4.

**PALABRAS CLAVE**

Expert Advisor, rentabilidad, mercados financieros, indicadores, Ichimoku, RSI, mercados tendenciales, optimización, parámetros, posición larga, posición corta, apertura de velas, programación, MetaEditor4, MQL4, MetaTrader4, backtesting.

**ABSTRACT**

An Expert Advisor has been developed with the objective of obtaining profitability in financial markets. To achieve this, the Ichimoku and RSI indicators have been used. These two indicators adapt to any market, but work especially well in trending markets. Therefore, we have focused on the development and optimization of the system in these specific markets.

To trade in different markets, the values of the parameters used must be taken into account. Based on them, the system will enter or exit. To enter a long position, the price must exceed the upper limit of the Ichimoku cloud and the RSI value must be below the overbought zone. To exit the long position, the price must cross the lower limit of the Ichimoku cloud and the RSI must be above the oversold zone.

The entry and exit conditions for short positions, are the same as for long positions, but reversed. That is, to enter a short position, the price must cross down the cloud and the RSI must be above the oversold zone. To exit the short position, the price must cross up the upper limit of the cloud and the RSI must be below the overbought zone.

It is important to note that the system is programmed to operate at the opening of candles. This means that the indicator values are taken from the previous candle. If we are not at the candle opening, the system will not operate.

For the development of the system, the MetaEditor4 programming environment and the corresponding MQL4 programming language have been used. The MetaTrader4 platform has been employed for optimization and backtesting.

**KEYWORDS**

Expert advisor, Profitability, Financial markets, Simple moving averages, RSI, Gold, Daily timeframe, Profitable set, Long operation, Short operation, Oversold, Overbought, Candle opening, Previous candle, Programming, MetaEditor4, MQL4, Optimization, Backtesting, MetaTrader4 platform.

**ÍNDICE**

**1** - [**INTRODUCCIÓN**](#introduccion) **5**

1.1 - [INTRODUCCIÓN Y CONCEPTO DE MERCADO FINANCIERO](#introduccion) **5**

1.2 - [CONCEPTOS BÁSICOS DE MERCADO FINANCIERO](#conceptosbasicos) **5**

1.3 - [INDICADORES EMPLEADOS EN EL SISTEMA](#indicadores) **7**

1.4 - [COMO HACER LA OPTIMIZACIÓN](#procesooptimizacion) **9**

**2** - [**OBJETIVOS**](#objetivos) **12**

2.1 - [PARÁMETROS OPTIMIZABLES Y CONDICIONES DEL SISTEMA](#objetivos)**\_\_\_\_\_\_\_\_\_12**

2.2 - [ENTRADAS Y SALIDAS DEL SISTEMA EN LOS MERCADOS](#entradassalidas)\_\_\_\_\_\_\_\_\_\_\_\_\_**13**

**3** - [**DESCRIPCIÓN INFORMÁTICA DEL SISTE**](#descripciondelsistema)**MA 16**

3.1 - [HERRAMIENTAS EMPLEADAS](#descripciondelsistema)**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_16**

3.2 - [DESCRIPCIÓN DEL SISTEMA](#sistema)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**\_\_17**

**4** - [**EXPERIMENTOS Y VALIDACIÓN**](#experimentosyvalidacion) **33**

4.1 - ELECCIÓN DE MERCADO Y TIMEFRAME**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_33**

4.2 - [CONFIGURACIÓN DE PARÁMETROS](#configuracionparam)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**33**

4.3- [PERIODO DE PRUEBA Y CONFIGURACIÓNES PREVIAS](#periododepruebayconfig)**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_37**

4.4 - [OPTIMIZACIÓN GENÉTICA](#optimizaciongenetica)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **38**

4.5 - [PRIMERA OPTIMIZACIÓN POR FUERZA BRUTA](#primeraopfuerzabruta)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**39**

4.6 - [SEGUNDA OPTIMIZACIÓN POR FUERZA BRUTA](#segundaopfuerzabruta)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**41**

4.7 - [TERCERA OPTIMIZACIÓN POR FUERZA BRUTA](#terceraopfuerzaburta)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**44**

4.8 - [BACKTEST FINAL](#backtestfinal)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**49**

4.9 - [INFORME FINAL](#informefinal)**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_51**

**5** - [**CONCLUSIONES**](#conclusiones) **54**

**6** - [**BIBLIOGRAFÍA**](#bibliografía) **58**

**7** - **APÉNDICE: CÓDIGO FUENTE DEL SISTEMA \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_60**

**1 – INTRODUCCIÓN**

Antes de entrar en detalle en el proceso llevado a cabo para el desarrollo del sistema, es importante poner en contexto el ámbito en el que se ha desarrollado este trabajo. Por lo tanto, se realizará una breve introducción para dar algunas nociones básicas a los lectores. Esto facilitará la comprensión posterior del sistema desarrollado y el porqué de las decisiones tomadas.

**1.1 - INTRODUCCIÓN Y CONCEPTO DE MERCADO FINANCIERO**

Los mercados financieros son una de las partes más importantes dentro de la economía global, ya que permiten la transferencia de fondos entre aquellos que tienen excedentes de capital y aquellos que necesitan financiamiento. En estos mercados se negocian una gran variedad de instrumentos financieros, como pueden ser acciones, bonos, divisas o materias primas.

Los mercados financieros pueden ser locales o internacionales y pueden ser de diferentes tipos, como el mercado de acciones, el mercado de bonos o el mercado de materias primas. Cada uno de estos mercados presenta sus propias características, regulaciones y participantes.

La evolución de los mercados financieros está vinculada a los cambios económicos y políticos que ocurren a nivel mundial, lo que los hace altamente volátiles y difíciles de predecir. Además, los mercados financieros están sujetos a constantes innovaciones y cambios en la tecnología, lo que hace que su análisis sea aún más complejo.

**1.2 - CONCEPTOS BÁS****ICOS DE UN MERCADO FINANCIERO**

En los mercados financieros existen dos tipos de operaciones: compra y venta. Cuando realizamos una operación de compra, lo hacemos con la esperanza de que el activo en cuestión suba de precio para venderlo a un precio superior al de compra y obtener un beneficio. En cambio, cuando entramos en una operación de venta, vendemos nuestro activo en busca de un precio más bajo al de venta, para volver a recomprarlo a un precio menor y así obtener beneficios y mantener nuestro activo.

Este tipo de operaciones, por supuesto, no se realizan de forma aleatoria, sino que son fruto de un análisis exhaustivo previo del mercado sobre el que se va a operar.

Todo lo explicado hasta este momento en este apartado, es lo que se conoce como trading.

Ahora que ya se ha explicado de forma breve las operaciones de compra y venta, debemos tener en cuenta algunos términos técnicos para este tipo de operaciones.

Al operar en un mercado financiero, es esencial analizar el precio en un período de tiempo específico, conocido como timeframe. Por ejemplo, cuando nos referimos a un timeframe diario, hablamos de la evolución del precio del activo durante los días.

Para realizar este análisis del precio en un timeframe determinado, utilizamos el gráfico de velas japonesas, donde cada vela representa una unidad de tiempo en el mercado. En otras palabras, si el timeframe es diario, las velas representan el rango de precios del activo en ese día.

El cuerpo de la vela japonesa representa la diferencia entre el precio de apertura y el precio de cierre de un activo en un periodo de tiempo determinado. Si el precio de cierre es mayor que el precio de apertura, la vela será alcista y su cuerpo estará coloreado generalmente en verde o blanco, mientras que, si el precio de cierre es menor que el precio de apertura, la vela será bajista y su cuerpo estará coloreado en rojo o negro.

Las velas también pueden tener lo que se conoce como mechas, que representan los valores atípicos del precio durante el periodo de tiempo analizado. Sin embargo, para nuestro sistema solo se toman en cuenta los precios de apertura y cierre de la vela.

El precio de entrada en una operación de compra se conoce como precio Ask, mientras que el precio de salida se denomina Bid. Por otro lado, en las operaciones de venta ocurre lo contrario, el precio de entrada es el Bid y el precio de salida es el Ask, que es el precio al que se puede comprar el activo.

Las operaciones de compra también se denominan largas y las operaciones de venta cortas. Cuando hablamos de entrar en largo, queremos decir que vamos a entrar en una operación de compra y cuando hablamos de entrar en corto, queremos decir que vamos a realizar una operación de venta.

La diferencia entre el precio de compra (Ask) y el precio de venta (Bid), es lo que se denomina Spread. Ask siempre será mayor que Bid.

Cuando nosotros lanzamos una operación al mercado, ya sea de compra o de venta, esta tiene que llegar al servidor donde se aloja nuestro mercado, por tanto, aunque este tiempo sea pequeño generalmente, puede existir una diferencia entre el precio al que nosotros abrimos una operación y el de la situación actual del mercado, esto es lo que se conoce como slippagge. Esto es importante tenerlo en cuenta sobre todo en nuestro caso que estamos desarrollando sistemas automáticos, ya que, si no se tiene en consideración, puede darse el caso que las operaciones se lancen a un precio diferente al del mercado, dando fallo al intentar abrirse y, por tanto, podamos perder una situación de mercado favorable para nuestros intereses.

**1.3 - INDICADORES EMPLEADOS EN EL SISTEMA**

Cuando deseamos operar en los mercados financieros, como se mencionó anteriormente, no es una actividad que llevemos a cabo de manera aleatoria. Más bien, tomamos la decisión de operar después de estudiar minuciosamente el mercado en el que vamos a operar.

Una de las herramientas que pueden ayudarnos a predecir el comportamiento de un mercado en particular son los indicadores. Existen numerosos tipos de indicadores, pero en esta sección solo se detallarán los utilizados para el desarrollo del sistema: Ichimoku y RSI.

El Relative Strength Index, también conocido como RSI, es un indicador utilizado para medir la fuerza relativa de un activo específico en función de su precio y volumen de operaciones en un período de tiempo determinado. Nos ayuda a detectar posibles cambios de tendencia en el mercado.

El RSI tiene un rango de valores que va de 0 a 100. El valor de 50 es el umbral que indica si el mercado se encuentra en tendencia alcista (por encima) o bajista (por debajo). Si el valor se encuentra por encima de 70, indica que el activo está sobrecomprado, lo que significa que es probable que el precio disminuya en el futuro. Por lo tanto, cuando el valor vuelva a caer por debajo de 70, sería una señal de entrada para una operación de venta.

Por otro lado, cuando el valor del RSI cae por debajo de 30, indica que el activo está sobrevendido, lo que significa que es probable que el precio aumente en el futuro. Así que cuando este valor vuelva a subir por encima de 30, será una señal de compra.

Es importante tener en cuenta estos valores del RSI para tomar decisiones informadas al operar en el mercado financiero.

El indicador de Ichimoku, es un indicador desarrollado en Japón, a mediados del siglo XX. Consta de 5 líneas que nos ayudaran a detectar zonas de soporte y resistencia, dirección de la tendencia o señales de entrada o salida.

A continuación, se mencionan las líneas que lo componen, así como la utilidad de cada de ellas:

1. La línea Tenkan-sen: también conocida como la línea rápida, es una herramienta útil para detectar la tendencia del mercado a corto plazo. Su valor se calcula mediante la suma del mínimo más bajo y el máximo más alto de las últimas nueve velas.
2. La línea Kijun-sen: también conocida como la línea lenta, es una herramienta útil para detectar la tendencia del mercado a medio plazo. Al igual que la línea rápida (Tenkan-sen), su valor se calcula a partir de la suma del mínimo más bajo y el máximo más alto de las velas anteriores. Sin embargo, en este caso se tienen en cuenta las últimas 26 velas.
3. La línea Senkou Span A: es una de las dos líneas que delimitan la nube de Ichimoku, tiene un valor que se define a partir de la suma de la línea rápida (Tenkan-sen) y la línea lenta (Kijun-sen) dividida entre dos. Esta línea es útil para predecir la tendencia del mercado a largo plazo.
4. La línea Senkou Span B: es la otra línea que forma la nube junto a la línea Senkou Span A, se calcula a partir de la suma del mínimo más bajo y el máximo más alto de las 52 velas anteriores. Al igual que la Senkou Span A, esta línea es útil para detectar la tendencia del mercado a largo plazo.
5. La línea Chinkou Span: también conocida como línea de retraso, es una línea retrasada que se muestra en el gráfico teniendo en cuenta el precio de cierre actual, pero 26 velas atrás. Esta línea es útil para predecir la tendencia del mercado a corto plazo.



Ilustración 1: Ejemplo práctico del indicador de Ichimoku con todas sus componentes. El gráfico pertenece al mercado ETHUSD con timeframe de 1 hora.

En la ilustración 1, se pueden ver las diferentes componentes del indicador de Ichimoku. A continuación, se explica cada una de ellas:

* La línea verde es el Chinkou Span
* La línea azul es el KijunSen
* La línea roja es el TenkanSen
* La línea discontinua blanca es Senkou Span B
* La línea discontinua naranja es el Senkou Span A
* El área comprendida entre el Senkou Span A y el Senkou Span B, es la nube. Cuando el valor del Senkou Span B, está por encima del Senkou Span A, la nube es de color blanca. En cambio, si el Senkou Span A está por encima, la nube será de color naranja.

En nuestro sistema no se utilizan las 5 líneas del indicador Ichimoku. En particular, la línea de retraso (Chinkou Span) no se utiliza en ningún momento. Nos centramos únicamente en las otras 4 líneas que definen la nube de Ichimoku. En función de cómo se comporte el precio alrededor de la nube, nuestro sistema actuará de una manera u otra.

**1.4 - C****OMO HACER LA OPTIMIZACIÓN**

Una vez ya hemos estudiado un mercado en cuestión, hemos desarrollado una estrategia para operar y hemos programado nuestro experto, llega el último, pero más importante paso: la optimización.

Cuando nosotros programamos nuestro experto, este tiene una serie de parámetros, que no son más que los valores que definen los indicadores empleados en su desarrollo. Pues bien, optimizar no es más que encontrar una combinación de valores que haga que nuestro sistema gane dinero.

Antes de empezar a optimizar, necesitamos tener históricos lo suficientemente largos del mercado que vamos a probar, lo ideal es que se pruebe en el mayor histórico posible, pero mínimo debe ser en un periodo de 10 años. Del mercado seleccionado debemos escoger también un timeframe, a mayor timeframe mayor probabilidad de encontrar un set rentable.

Una vez tenemos el histórico del mercado que vamos a probar y hemos seleccionado el timeframe, ya podemos empezar con la optimización.

En primer lugar, debemos escoger un rango de valores para los parámetros de nuestro sistema, este rango de valores debe ser adecuado a la estrategia que vamos a seguir y por supuesto deben ser valores con sentido, por ejemplo, los valores 30 y 70 del RSI que son los límites de sobreventa y sobrecompra respectivamente.

Antes de continuar, se van a introducir los conceptos de optimización genética y de optimización por fuerza bruta, a los que se hará mención con frecuencia en este documento y que, por tanto, son vitales para su entendimiento.

La optimización genética consiste en la prueba de diferentes valores de los parámetros del sistema dentro de los rangos establecidos en el primer paso, con el fin de encontrar la mejor combinación de valores para nuestra función objetivo. La función objetivo no es más que lo que buscamos con la optimización; máximo balance, mínimo drawdown, profit factor…

En nuestro caso, vamos a buscar el máximo balance con nuestras optimizaciones y no vamos a profundizar en el resto, pero está bien saber que podemos optimizar basándonos en otros objetivos que no sean buscar el máximo balance.

La optimización genética se basa en la teoría de la selección natural, se empieza por una combinación inicial y esta va mutando poco a poco, dejando únicamente las combinaciones de valores interesantes en función a nuestro objetivo y eliminando aquellas que no llevan a ninguna parte.

La optimización por fuerza bruta, por su parte, realiza una búsqueda exhaustiva de todos los posibles valores de los parámetros del sistema, probando todas las combinaciones posibles de estos con el fin de encontrar la combinación que maximice la rentabilidad.

Por tanto, debemos tener en cuenta que el consumo de tiempo y de recursos por parte de la optimización por fuerza bruta, será mucho mayor que la de la optimización genética. Sin embargo, en ambos casos debemos tener en cuenta que, aunque encontremos una combinación de valores que nos proporcione beneficios, no quiere decir que en el mercado real las vaya a producir también, ya que las condiciones no serán las mismas.

Ahora que ya se entienden los conceptos de optimización genética y por fuerza bruta, podemos continuar explicando los pasos de la optimización. Tras establecer el rango de valores de nuestros parámetros, ahora debemos seleccionar aproximadamente las dos terceras partes de nuestro histórico y realizar una optimización genética de todos los parámetros del sistema a la vez.

De los resultados obtenidos en este paso, cogeremos aquellos sets que den beneficios y que tengan una combinación de valores muy diferente entre sí, para probar todo el posible rango de valores.

De cada set seleccionado en el paso anterior, haremos una optimización por fuerza bruta de todos los parámetros de nuestro sistema en las dos terceras partes más antiguas del histórico, empezando por los dos parámetros más relacionados e importantes.

A continuación, se repite el paso anterior para el resto de parámetros hasta terminar de optimizar todos. En caso de que no podamos optimizar dos a la vez, porque nuestro sistema tenga un número de parámetros impar, una vez optimizados el resto de parámetros, se optimizaría este último, dándonos así los sets válidos con los que haremos las pruebas finales.

En las optimizaciones por fuerza bruta, los sets considerados como válidos, serán aquellos que se encuentren en el centro de mesetas. Una meseta no es más que una zona del gráfico, donde a medida que los valores de los parámetros varían, los beneficios cambian de forma suave y progresiva.

Llegados a este punto, si no tenemos ningún set rentable, debemos cambiar de timeframe o cambiar de mercado y repetir los pasos en busca de un set que si lo sea.

En caso contrario, ya solo nos queda probar nuestros sets en la última parte de nuestro histórico. Para considerar en esta última parte que un set es válido, la gráfica de evolución de balance, además de dar beneficios, debe ser lo más lineal posible, es decir, que el beneficio vaya poco a poco aumentando, además, el número de operaciones en esta parte debe ser mayor de 50. Si encontramos un set que lo cumpla, ese set será rentable y nos lo guardaremos, si no encontramos ninguno válido, cambiamos de mercado o de timeframe y volvemos a probar.

Una vez optimizado el sistema y encontrado uno o varios sets rentables, lo ideal, sería probar el sistema con una cuenta demo durante al menos 3 meses, para ver cómo se comporta en condiciones de mercado reales y ahí decidir si lo probamos con dinero real o no.

Como se ha podido ver, la optimización es la parte más importante del desarrollo del sistema, no podemos desarrollar un sistema y lanzarlo a un mercado con condiciones reales sin haberlo probado previamente, ya que las probabilidades de perder son bastante altas.

**2 – OBJETIVOS**

El objetivo de este proyecto es desarrollar un sistema de trading automático (EA) que nos proporcione beneficios operando de forma autónoma en los mercados financieros, sin tener que estar nosotros detrás estudiando los mercados constantemente. Con esto, eliminamos el factor humano que nos puede jugar malas pasadas, ya que puede que por un sentimiento de miedo podamos cerrar operaciones en un momento que no es conveniente, haciendo que perdamos dinero por dudar de nuestra estrategia o reduciendo nuestros beneficios cerrando operaciones antes de lo que deberíamos.

**2.1- PARÁMETROS OPTIMIZABLES Y CONDICIONES DEL SISTEMA**

El sistema desarrollado emplea dos indicadores diferentes, el oscilador RSI (Relative Strength Index) y el indicador de Ichimoku. Como hemos visto en el apartado anterior, cada uno de estos indicadores tiene una serie de valores a tener en cuenta, estos serán los parámetros optimizables de nuestro sistema. Para el caso del RSI, optimizaremos su periodo de cálculo, valor de sobrecompra y valor de sobreventa. Para el indicador de Ichimoku, optimizaremos los periodos de las líneas rápida y lenta (Tenkan-sen y Kijun-sen), el Senkou Span A y el Senkou Span B.

Por tanto, los parámetros optimizables de nuestro sistema, serán los siguientes:

* Periodo Tenkan-Sen
* Periodo Kijun-Sen
* Senkou Span B
* Periodo RSI
* Sobrecompra
* Sobreventa

En función de sus valores y su evolución nos determinarán las entradas y salidas del sistema. Para la entrada en operaciones, ya sea corta o larga, tendremos en cuenta el valor de los parámetros del sistema en la vela anterior, pero solo operaremos cuando nos encontremos en la apertura de velas.

El sistema únicamente realiza un único tipo de operación a la vez, es decir, si se abre una operación de compra, mientras esta esté abierta, aunque se cumplan las condiciones de mercado para abrir más operaciones de compra, estas no se abrirán, ya que hay una operación abierta de ese mismo tipo. Para las operaciones de venta se aplica lo mismo.

Además, el sistema controla los valores del periodo de la línea rápida (Tenkan-sen) y línea lenta (Kijun-sen). Si el valor del periodo de la línea lenta es más pequeño que el de la línea rápida, el sistema no operará, pues no tendría mucho sentido.

**2.2 - ENTRADAS Y SALIDAS DEL SISTEMA EN LOS MERCADOS**

Una vez que los parámetros de nuestro sistema y sus limitaciones para operar han sido definidos, se presentan a continuación las entradas y salidas del mercado en función de los valores de dichos parámetros.

* Entraremos largo si el precio cruza hacia arriba el límite superior de la nube y el valor del RSI se encuentra por debajo de la sobrecompra
* Entraremos en corto si el precio cruza hacia abajo el límite inferior de la nube y el valor del RSI se encuentra por encima de la sobreventa
* Saldremos del corto si el precio cruza hacia arriba el límite superior de la nube y además el valor del RSI se encuentra por debajo de sobrecompra
* Saldremos del largo si el precio cruza hacia abajo el límite inferior de la nube y además el valor del RSI se encuentra por encima de la zona de sobreventa

Para determinar el cruce con la nube, nos fijamos en el precio de 1 y 2 velas más atrás, es decir, para considerar cruce, el precio dos velas atrás debe estar por encima/debajo del límite inferior/superior de la nube y el precio una vela atrás deberá estar por debajo/encima del límite superior/inferior de la nube. Si esto se cumple, se abrirá la operación correspondiente en la apertura de la siguiente vela.

A continuación, se mostrarán una serie de ilustraciones que ejemplificarán las entradas y salidas del sistema en el mercado. Para ello tomaremos los siguientes valores para nuestros parámetros:

* Periodo Tenkan-Sen: 9
* Periodo Kijun-Sen: 26
* Senkou Span B: 52
* Periodo RSI: 14
* Sobrecompra: 70
* Sobreventa: 30



Ilustración 2: Operación de compra en el mercado ETHUSD con timeframe de 1 hora

En la ilustración 2 se muestra un ejemplo de entrada en largo al mercado. En la parte inferior del gráfico, se observan las líneas azules, las cuales indican el valor del RSI en cada momento. Las líneas horizontales verdes representan los límites de sobrecompra (línea superior, valor 70) y sobreventa (línea inferior, valor 30). En la parte superior se encuentra el gráfico de precio del mercado y la nube de Ichimoku. Se han ocultado el resto de componentes del indicador de Ichimoku para dar claridad al gráfico, ya que no son necesarios en este caso.

Las líneas horizontales rojas señalan la entrada y salida del mercado (la línea más a la izquierda y la más a la derecha, respectivamente). En este caso, se entra en largo cuando el precio cruza el límite superior de la nube, además, el valor del RSI no supera el límite de sobrecompra. Posteriormente, se sale de la operación larga cuando el precio cruza hacia abajo el límite inferior de la nube y el valor del RSI se encuentra por encima del límite de sobreventa.



Ilustración 3: Operación de venta en el mercado ETHUSD timeframe de 1 hora

En la ilustración 3 se muestra la entrada y la salida de una operación de venta. Para entrar al mercado hemos cruzado hacia abajo el límite inferior de la nube, además, el valor del RSI se encuentra por encima de la sobreventa, por tanto, abrimos en corto. Saldremos de la operación en corto más adelante, cuando el precio vuelva a cruzar el límite superior de la nube hacia arriba y el valor del RSI esté por debajo de la sobrecompra.

**3 - DESCRIPCIÓN INFORMATICA DEL SISTEMA**

Ahora que ya tenemos claros algunos conceptos básicos de los mercados, hemos definido las condiciones de entrada y salida de nuestro sistema en el mercado, las hemos comprendido con ejemplos prácticos en mercado real, hemos establecido las restricciones y condiciones del sistema y hemos identificado los parámetros a optimizar, ya podemos comenzar con el desarrollo de nuestro experto.

**3.1 - HERRAMIENTAS EMPLEADAS**

Antes de entrar en el desarrollo del sistema en sí, es bueno que comprendamos y nos familiaricemos con las herramientas que vamos a emplear, en este caso son dos: MetaTrader4 y MetaEditor.

MetaTrader4 (MT4) es una plataforma de trading en línea donde personas de todo el mundo, independientemente de su nivel, pueden comprar y vender una gran cantidad de activos distintos, que van desde materias primas y acciones a bonos o divisas.

MT4 nos ofrece además una gran cantidad de alternativas para analizar los mercados, contando con distintos tipos de gráficas avanzadas, indicadores de trading de los cuales solo nos tenemos que preocupar de configurar sus parámetros y backtesting que nos permitirá realizar pruebas en distintos mercados con distintos timeframes. Esta última parte es la más importante de MT4 para nosotros, ya que aquí probaremos el sistema desarrollado.

MetaEditor es el entorno de desarrollo donde programaremos nuestro sistema, para ello se emplea su propio lenguaje de programación MQL, un lenguaje con una sintaxis parecida a C.

La parte más importante para nosotros, es que MetaEditor nos da la posibilidad de desarrollar asesores expertos, que al final es el objeto de este trabajo.

**3.2 - DESCRIPCIÓN DEL SISTEMA**

Una vez conocemos las herramientas necesarias, ya podemos comenzar a desarrollar nuestro sistema. A continuación, se mostrará el código fuente del sistema desarrollado párrafo a párrafo explicando la funcionalidad de cada parte.

#property strict

#define COPYRIGHT "Sergio Bolaños GII"

#define VERSION "1.0"

#define FECHA 24042023

#define NOMBRE "IchimokuRSI"

#property copyright COPYRIGHT

#property version VERSION

Esta es la parte inicial del programa, aquí definimos algunas constantes (define) y propiedades del sistema (property). Property strict es una directiva que hace al compilador ser más estricto con nuestro código fuente, no nos va a permitir saltarnos ninguna regla de tipado, generando errores, además, no vamos a poder realizar operaciones entre tipos de datos distintos. Esto lo sabemos gracias al curso impartido por el tutor de este trabajo [3].

Las 4 define componen los datos del sistema programado. El primero es copyright, donde pondremos el nombre del autor, nuestro nombre en este caso. El segundo indica la versión del sistema, en este caso 1.0, ya que es la primera versión del sistema y no se ha actualizado ni modificado. El tercero indica la fecha de desarrollo del sistema, 24042023 indica 24/04/2023. Por último, el nombre, aquí pondremos el nombre del sistema desarrollado.

//Parámetros Optimizables del EA

//Periodo Tenkan

input uint PeriodoTenkanSen = 9;

//Periodo Kinjun

input uint PeriodoKinJunSen = 26;

//límite Senkou SpanB

input uint PeriodoSenkouSpanB = 52;

//Periodo para el RSI

input uint PeriodoRSI = 14;

//Umbral de Sobrecompra

input uint Sobrecompra = 70;

//Umbral de Sobreventa

input uint Sobreventa = 30;

A continuación, tenemos la zona de declaración de parámetros del sistema. Los parámetros se dividen en dos partes: los parámetros optimizables y los no optimizables. La parte que se muestra justo encima de este párrafo, hace referencia a los parámetros optimizables.

Los parámetros optimizables son aquellos que, como su propio nombre dice, vamos a querer optimizar, buscando una combinación de valores que nos lleve a generar beneficios (esto se detalla más adelante en este documento). Los parámetros no optimizables son aquellos que vamos a dejar fijos en todo momento y su valor se mantendrá en el valor inicial.

Todas las variables están declaradas con el término input (indica que son variables de entrada) y su tipo de dato. El tipo uint indica número entero sin signo.

Por tanto, como parámetros optimizables tenemos:

* PeriodoTenkanSen (línea rápida): se inicializa a 9 por defecto, periodo de cálculo de la línea rápida.
* PeriodoKijunSen (línea lenta): se inicializa a 26 por defecto, periodo de cálculo de la línea lenta.
* PeriodoSenkouSpanB: se inicializa a 52 por defecto, contiene el periodo, con el que calcularemos uno de los límites de la nube.
* PeriodoRSI: se inicializa a 14 por defecto, periodo de cálculo para el RSI
* Sobrecompra: se inicializa a 70 por defecto, valor límite que marca la zona de sobrecompra del RSI.
* Sobreventa: se inicializa a 30 por defecto, valor límite que marca la zona de sobreventa del RSI.

//Parametros no Optimizables del EA

//No optimizar

//identificador de operación

input uint Magicnumber = FECHA;

//lotaje por operación

input double lotaje = 0.1;

En la parte de parámetros no optimizables tenemos lotaje, que indica el número de lotes por operación, este valor se mantendrá fijo a 0.1, pues no nos interesa optimizar, a mayor valor mayor beneficio/perdida y a menor valor menor beneficio/perdida. El otro parámetro es el magicnumber, no es más que un número identificativo de operación. Este valor va a permitir al sistema encontrar una operación en concreto.

//Variables globales

//Indica si hay error en los parámetros

bool ErrParam = false;

//identificador operación corta,

//-1 si no hay ninguna

int idcorta;

//identificador operación larga,

//-1 si no hay ninguna

int idlarga;

//tiempo de la última vela procesada

//de esta manera el EA será OpenPrice

datetime TiempoUltimaVelaProcesada;

Ahora tenemos la sección de variables globales, para nuestro sistema nos hacen falta 4: ErrParam es una variable booleana que indica si hay algún error en los parámetros (más adelante se detallan los posibles errores), por defecto se inicializa false, pues se supone que todo va a ir bien de inicio. Las variables idcorta e idlarga son los identificadores de operación de las operaciones cortas y largas respectivamente, recibirán el valor de otra función que veremos más adelante, si este valor es -1 es porque no hay ninguna operación de ese tipo abierta. Por último, tenemos TiempoUltimaVelaProcesada, que es una variable de tipo datetime. Datetime es un tipo de datos que permite representar fechas y horas en el formato de mql por defecto, representadas por un número. Este número, serán los segundos transcurridos desde el 1 de enero de 1970. Esta variable, la usaremos para controlar que el sistema opere en la apertura de las velas y contendrá el tiempo transcurrido hasta la última vela procesada.

/\*se ejecuta al iniciar el EA

comprueba si hay error en los parámetros

y si hay alguna operación abierta

\*/

int OnInit(){

//error en los parámetros

ErrParam = PeriodoKinJunSen<PeriodoTenkanSen;

//inicialmente 0, ya que no se ha procesado

//ninguna vela todavía

TiempoUltimaVelaProcesada = 0;

//-1 si no hay operaciones abiertas con ese magic

//si no tendrá el id de la operación actual

idlarga = OperacionAbierta(Magicnumber);

//-1 si no hay operaciones abiertas con ese magic

//si no tendrá el id de la operación actual

idcorta = OperacionAbierta(Magicnumber);

//si hay error en parámetros

//se muestra por consola

if(ErrParam)

printf("Error en los parámetros , el EA no operará");

return(INIT\_SUCCEEDED);

}

La función OnInit es una función que se ejecuta una sola vez y al inicio de nuestro sistema, su valor de retorno dependerá del valor de la constante predefinida init\_succeeded, esta devolverá 0 si el inicio ha sido correcto.

Dentro de la función OnInit inicializaremos las variables globales del sistema: ErrParam, comprobará el valor establecido para los periodos de la línea lenta (Kijun-sen) y línea rápida (Tenkan-sen). Se inicializará a true en caso de que el periodo de la línea lenta sea menor al de la rápida, pues no tiene mucho sentido y false en caso contrario. TiempoUltimaVelaProcesada se inicializa a 0, pues estamos arrancando el sistema, por lo que aún no se ha procesado nada. idcorta e idlarga dependerán del valor que devuelva la función OperacionAbierta, que, aunque se detallara más adelante, nos permitirá saber si había algún tipo de operación abierta al iniciar el sistema. Por último, si hay un error en los parámetros se mostrará un mensaje por consola indicándolo.

//Se ejecuta en cada tick

void OnTick(){

//si hay error en los parametros o

//no nos encontramos en la apertura de la vela

//no hacemos nada

if(ErrParam || Time[0] <= TiempoUltimaVelaProcesada) return;

//si no hay error en los parametros y

//nos encontramos en la apertura de la vela

//podemos operar

if(!ErrParam && Time[0] > TiempoUltimaVelaProcesada){

La parte anterior hacía referencia a la fase de definición de variables e inicialización del sistema. Ahora entramos en la función principal, la función OnTick, donde se definirán las entradas y salidas del sistema, así como otros aspectos importantes de este.

La funcion OnTick se ejecuta una vez por cada tick. Un tick, no es más que un cambio en el precio del mercado en el que se esté operando, por cada tick realizado, se comprobará si existe error en los parámetros o si nos encontramos en apertura de una vela, en caso de darse una de las dos condiciones, no operaríamos. Al contrario, si no se cumplen ninguna de las dos condiciones, significará que no hay error en los parámetros y que nos encontramos en la apertura de una vela, por tanto, podremos operar.

//variables para guardar el límite superior

//e inferior de la nube

//en cada caso, esto lo necesitamos

//ya que nuestro sistema tendrá en cuenta

//si el precio está por encima o por debajo

double Lsnube, Linube;

//variables que contendran el límite superior

//y límite inferior de la nube de ichimoku

double SenkouSpanA, SenkouSpanB;

//variable que contiene el precio de

//cierre de una y dos velas atras

//respectivamente

double PrecioCierre, PrecioCierre1;

//variable que contiene el valor del RSI

//en la vela anterior

double RSIAnterior;

//Variables para el cálculo del slipagge que vamos

//a permitir en nuestras operaciones

int Slipagge;

double Spread;

Ahora declaramos las variables que contendrán los valores de los indicadores empleados para el sistema. Lsnube y Linube contendrán el límite superior e inferior de la nube respectivamente, pero en la vela anterior a la actual. SenkouSpanA y SenkouSpanB contendrán los límites de las nubes, no son lo mismo que las dos variables anteriormente citadas, SenkouSpanA y SenkouSpanB guardan el valor de los límites de la nube, Lsnube y Linube guardarán cuál de los dos valores es mayor (Lsnube) y menor (Linube). PrecioCierre y PrecioCierre1, contendrán el precio de cierre del mercado en una y dos velas atrás respectivamente. RSIAnterior, contendrá el valor del RSI en la vela anterior a la de apertura. Por último, las variables slipagge y spread contendrán los valores del slipagge que permitiremos y el spread en el momento de operar.

//Declaramos las variables que

//van a contener las condiciones

//True si precio > nube

bool condAlcista1 = false;

//True si RSI por debajo de sobrecompra;

bool condAlcista2 = false;

//True si precio < nube;

bool condBajista1 = false;

//True si RSI por encima de sobreventa;

bool condBajista2 = false;

Ahora se declaran 4 variables booleanas, que nos permitirán saber qué condiciones se están cumpliendo y cuáles no en el sistema. Hay dos para las condiciones alcistas y dos para las bajistas. Las condiciones de cada una de estas, se encuentran detalladas en los comentarios del código.

//Si se puede abrir alguna operación

//calculamos las Ichimoku y RSI

if(idlarga == -1 || idcorta == -1){

//Calculamos los valores necesarios

//cálculo del precio de cierre de

//una y dos velas atrás

PrecioCierre = Close[1];

PrecioCierre1 = Close[2];

//Calculo de los límites de la nube

//SenkouSpanA y SenkouSpanB

//y del RSI en la vela anterior

SenkouSpanA = iIchimoku(NULL,0,PeriodoTenkanSen,

PeriodoKinJunSen,PeriodoSenkouSpanB,

MODE\_SENKOUSPANA,1);

SenkouSpanB = iIchimoku(NULL,0,PeriodoTenkanSen,

PeriodoKinJunSen,PeriodoSenkouSpanB,

MODE\_SENKOUSPANB,1);

RSIAnterior = iRSI(NULL,0,PeriodoRSI,PRICE\_CLOSE,1);

Comprobamos que se puede iniciar algún tipo de operación, esto ocurrirá cuando no haya una operación larga o una corta abierta (idlarga o idcorta valen alguno de los dos -1), si esto pasa inicializamos el valor de los indicadores del sistema. Para los precios de cierre llamamos a la función Close, que nos dará el precio de cierre de la vela que le pasemos entre corchetes (en este caso 1 y 2, porque queremos, el valor de 1 y 2 velas atrás). Para el cálculo de los límites de la nube se utiliza la función predefinida iIchimoku. Basándonos en la documentación de mql4 para este indicador [[14],](#bibliografía) sabemos que tendrá como parámetros:

1. Mercado: mercado sobre el que se calcula, en este caso es null que indica el mercado que seleccionemos en Metatrader.
2. Timeframe: es el timeframe que emplearemos, al ponerlo a 0 será el timeframe que seleccionemos nosotros.
3. PeriodoTenkanSen: Periodo de cálculo de la línea rápida.
4. PeriodoKijunSen: Periodo de cálculo de la línea lenta.
5. MODE: aquí diremos que valor queremos calcular, si el SenkouSpanA (MODE\_SENKOUSPANA) o el SenkouSpanB (MODE\_SENKOUSPANB).
6. Vela de cálculo: aquí indicamos sobre qué vela calculamos el valor, en nuestro caso una vela atrás.

Ahora nos quedaría calcular el valor del RSI para la vela anterior a la actual, para ello y basándonos en la documentación de mql4 para este indicador [[8]](#bibliografía), le hemos pasado los siguientes parámetros:

1. Mercado: mercado donde se calcula, siendo null es el mercado que seleccionemos.
2. Timeframe: que será el que seleccionemos nosotros y no ninguno en concreto.
3. PeriodoRSI: es el periodo que se emplea para calcular el RSI.
4. Precio: Precio que se toma de la vela, en este caso es PRICE\_CLOSE, pues nos interesa el precio de cierre.
5. Vela de cálculo: aquí se indica la vela sobre la que se calcula el valor del indicador, 1 en este caso, pues queremos su valor en la vela anterior.

//Averiguamos las condiciones que se dan

//Antes debemos calcular la nube, la nube

//viene delimitada por el SenkouSpanA y

//el SenkouSpanB, dependiendo cual este por

//encima sabremos cual es el limite

//superior o inferior

if(SenkouSpanA > SenkouSpanB){

Lsnube = SenkouSpanA;

Linube = SenkouSpanB;

}else if(SenkouSpanA < SenkouSpanB){

Lsnube = SenkouSpanB;

Linube = SenkouSpanA;

}else{

Lsnube = SenkouSpanB;

Linube = Lsnube;

}

Aquí calcularemos el límite superior e inferior de la nube. Dependerá del valor del SenkouSpanA y el SenkouSpanB:

* Si el valor del SenkouSpanA es mayor al del SenkouSpanB, el límite superior será el SenkouSpanA y el inferior SenkouSpanB.
* Si el valor del SenkouSpanB es mayor al del SenkouSpanA, el límite superior será el SenkouSpanB y el inferior SenkouSpanA.
* En caso (poco probable) de que los dos valores sean iguales, el límite superior e inferior será el mismo (tomando el SenkouSpanA o SenkouSpanB indistintamente)

//Comprobamos si se cumplen

//las condiciones de entrada del sistema

//condiciones alcistas

condAlcista1 = PrecioCierre1 < Lsnube

&& PrecioCierre > Lsnube;

condAlcista2 = RSIAnterior < Sobrecompra;

//condiciones bajistas

condBajista1 = PrecioCierre1 > Linube

&& PrecioCierre < Linube;

condBajista2 = RSIAnterior > Sobreventa;

Aquí comprobamos si se cumplen las condiciones de entrada al sistema

//Calculamos el Slipagge máximo que vamos

//a permitir en nuestras operaciones

//El slipagge que vamos a permitir como máximo será 2 veces

//el spread del mercado en el que se opere

//Así damos margen a que si alguna operación entra

//con unas condiciones diferentes a las que se mando

//entre sin problema y no de errores

Spread = (Ask - Bid)\*(MathPow(10 , Digits));

Spread = NormalizeDouble(Spread ,Digits);

Slipagge = (int)Spread\*2;

Aquí se calcula el spread permitido en nuestras operaciones y el slippage. El porqué de usar estos valores, viene descrito en los comentarios del propio código, que se encuentran justo encima del cálculo de las variables.

//si se cumplen las dos condiciones alcistas definidas

//y no hay otra operación larga abierta previamente

//abrimos una operación larga

if((condAlcista1 && condAlcista2) && idlarga == -1)

idlarga = AbrirOperacion(OP\_BUY,lotaje,Slipagge,Magicnumber);

//si se cumplen las dos condiciones bajistas definidas

//y no hay una operación corta abierta previamente

//abrimos una operación corta

if((condBajista1 && condBajista2) && idcorta == -1)

idcorta = AbrirOperacion(OP\_SELL,lotaje,Slipagge,Magicnumber);

Aquí se comprueba si se cumplen las condiciones bajistas o alcistas que nos permitan abrir una operación, si no es así, pues en este tick no se podrá abrir operación y se volverá a probar en el siguiente.

Para las operaciones de compra debe cumplirse alguna de las dos condiciones alcistas definidas anteriormente, además, no debe existir otra operación de ese mismo tipo abierta antes. Para las operaciones de venta se aplica el mismo concepto, solo que con las condiciones bajistas en vez de las alcistas.

Si se puede abrir algún tipo de operación, los valores de idlarga e idcorta se inicializarán al valor devuelto por la función abriroperacion que se verá más adelante, pero que básicamente abrirá una operación teniendo en cuenta los parámetros que le pasemos.

//si existe alguna operación abierta

//entonces cerramos la operación que corresponda

//según las condiciones del mercado en ese momento

if(idcorta != -1 || idlarga != -1){

//si se cumplen las dos condiciones

//alcistas que tenemos definidas

//y hay una operación corta abierta

//cerramos la operación corta

if((condAlcista1 && condAlcista2) && idcorta != -1)

idcorta = CerrarOperacion(Ask , lotaje , idcorta);

//si se cumplen las dos condiciones

//bajistas que tenemos definidas

//y hay una operación larga abierta

//cerramos la operación larga

if((condBajista1 && condBajista2) && idlarga != -1)

idlarga = CerrarOperacion(Bid , lotaje , idlarga);

}//fin del if(No hay operaciones abiertas)

Si hay alguna operación abierta sea del tipo que sea, comprobamos si se cumplen las condiciones para cerrarla. Para cerrar una operación bajista, deben cumplirse las condiciones alcistas y para cerrar una operación alcista, deben cumplirse las condiciones bajistas.

Idcorta e idlarga recibirán el valor de la función cerraroperacion, que devolverá un -1 si la operación se ha cerrado indicando que ya no hay una operación de ese tipo abierta, si no, mantendrán su valor anterior y la operación no se cerrará.

//Abre una operación y comprueba si hay alguna operación fantasma

//Parámetros: Lotaje: El lotaje de la operación

// Tipo: El tipo de operación (compra o venta)

// SM: Máximo slipagge permitido

// idmagic: El magicnumber de esa operación

//Retorno: Devuelve el identificador de la operación

//si la ha podido abrir sin problemas

//o -1 en caso de que exista algún tipo de error.

//Precondición: Para abrir una operación, ya sea compra o venta,

//no debe haber abierta una operación de ese mismo tipo y

//deben cumplirse las condiciones necesarias en el mercado

//para poder abrirlas

int AbrirOperacion(int tipo,double lotaje,int SM,int idmagic){

double apertura;//precio de apertura de la operacion

int id;//valor de retorno al abrir la operacion

//si la operación es una compra (tipo == OP\_BUY)

//el precio de apertura es el Ask

//si la operación es una venta (tipo == OP\_SELL)

//el precio de apertura es el Bid

if(tipo == OP\_BUY){

apertura = Ask;

}else{

apertura = Bid;

}

//abre una operación y guarda el valor en la variable id

//si se abre correctamente se guarda el valor de OrderSend

//si no se abre correctamente devuelve -1 y

//se vuelve a intentar en el siguiente tick

id = OrderSend(Symbol(), tipo,lotaje,apertura,SM,0,0,NULL,idmagic);

//esperamos por una posible operación fantasma

if(id == -1) GetOperacionFantasma(GetLastError());

return id;

}//Fin AbrirOperacion

La primera de las funciones auxiliares es la función abriroperacion, tanto los parámetros de entrada como las distintas acciones realizadas en esta función, se encuentran detallados en los comentarios, pero aun así vamos a explicarlos.

Las variables apertura e id guardan el precio de apertura y el identificador de la operación respectivamente. La variable apertura tendrá el precio Ask si la operación es de compra, es decir, si el parámetro tipo de esta es OP\_BUY, si no será operación de venta y, por tanto, el precio de apertura será el Bid.

A continuación, intentamos abrir una operación mediante la función predefinida OrderSend basándonos en la documentación de mql4 referente a esta función [[9]](#bibliografía), sabemos que devolverá el valor -1 si no se abre la operación o el identificador asignado a la operación en caso de sí abrirla.

Los parámetros que recibe serían los siguientes:

1. Mercado: es el símbolo del mercado donde se opera, symbol () indica el mercado que seleccionemos.
2. Tipo de operación: operación de compra o venta.
3. Lotaje: lotaje por operación, que ya sabemos que se mantiene fijo en 0.1 para nuestro sistema.
4. Slipagge: es el slipaggeMaximo que vamos a aceptar, este valor se calcula en la función ontick como ya se ha explicado anteriormente.
5. StopLoss: 0 indica que está desactivado
6. Take profit: 0 indica que está desactivado.
7. Comentario: es el comentario que queremos mostrar por consola cuando se abre la operación, en nuestro caso nada por eso es Null.
8. Magicnumber: es el magicnumber para la operación.

Antes de retornar el valor del identificador se comprobará, en caso de ser -1, el tipo de error obtenido(GetLastError). Este error será gestionado por la función getoperacionfantasma que se verá a continuación.

//Controla los tipos de error que pueden ocurrir

//al abrir una operación

//Parámetros:

// error: código de error que se genera cuando

// no se puede meter una operación

//Precondición: No se ha podido abrir una operación

//de forma satisfactoria (id operación = -1)

void GetOperacionFantasma(int error){

//si no hay errores no hacemos nada

if(error == ERR\_NO\_ERROR) return;

//si ocurre un error de los siguientes

//es que no ha ocurrido una operación fantasma

//y por tanto no podemos hacer nada

if (error==ERR\_INVALID\_TRADE\_PARAMETERS ||

error==ERR\_NOT\_ENOUGH\_RIGHTS || error==ERR\_TOO\_FREQUENT\_REQUESTS

|| error==ERR\_ACCOUNT\_DISABLED || error==ERR\_INVALID\_ACCOUNT ||

error==ERR\_INVALID\_PRICE || error==ERR\_INVALID\_STOPS ||

error==ERR\_INVALID\_TRADE\_VOLUME || error==ERR\_MARKET\_CLOSED ||

error==ERR\_TRADE\_DISABLED || error==ERR\_NOT\_ENOUGH\_MONEY ||

error==ERR\_PRICE\_CHANGED || error==ERR\_OFF\_QUOTES ||

error==ERR\_REQUOTE || error==ERR\_LONG\_POSITIONS\_ONLY\_ALLOWED ||

error==ERR\_TOO\_MANY\_REQUESTS || error==ERR\_TRADE\_MODIFY\_DENIED ||

error==ERR\_TRADE\_TOO\_MANY\_ORDERS ||

error==ERR\_TRADE\_HEDGE\_PROHIBITED ||

error==ERR\_TRADE\_PROHIBITED\_BY\_FIFO)

return;

//Si no se ha producido ninguno de los errores anteriores

//esperamos a que la operación aparezca

//en MetaTrader y la podamos recuperar en uno de los siguientes ticks

//esperamos 5 ticks, que solo llegaran si hay conexión

for (int i=0; i<5; i++) {

//Se espera a que llegue un nuevo tick

//esta comprobación se hace cada 2 segundos para no saturar

while (RefreshRates()==false) Sleep(2000);

Sleep(1000); //Esperamos en el este tick

}

}//Fin GetOperacionFantasma

La función GetoperacionFantasma recibe el valor identificativo del error, pues todos los errores tienen asignado un número determinado.

Primero se comprueba si el error es del tipo ERR\_NO\_ERROR, que realmente indica que no hay error, si esta condición se cumple retornamos, pues no ha habido error de ningún tipo; Segundo, se comprueba si el error coincide con alguno de los posibles errores más comunes al abrir una operación; Tercero y último, en caso de no ser ninguno de los errores anteriores se comprueba si es una operación fantasma. Una operación fantasma es una operación que se ha abierto, pero no se puede ver por problemas con el metatrader, como pueden ser problemas de conexión, para ello, esperaremos 5 ticks, donde cada tick se esperará un periodo de dos segundos, dentro de cada tick se esperará un segundo, esto se hace para dar margen a que aparezca la operación. En caso de que pasados los 5 ticks la operación no aparezca significará que no se ha abierto.

//Cierra una operación abierta

//Parámetros: salida: Precio de salida de la operación

// si es una venta será precio Ask si no Bid

// lotaje: Lotaje de la operación

// id: El identificador de la operación

//Retorno: devolverá -1 en caso de que se cierre la operación

// permitiendo así que se puedan abrir nuevas operaciones

// o devolverá el id de la operación en caso de que

// no se pueda cerrar (es decir se mantiene abierta)

//Precondicion: Para poder cerrar una operación es necesario que

// exista algún tipo de operación abierta

int CerrarOperacion(double salida , double lotaje , int id){

int v = id;//valor de retorno al cerrar la operación

bool cerrar = OrderClose(id , lotaje , salida , 0);

//si la operación se ha encontrado y

//se ha podido cerrar v = -1

if(cerrar) v = -1;

return v;

}//Fin de CerrarOperacion

La función CerrarOperacion se llamará cuando queramos cerrar una operación y se den las condiciones para ello. Es una función muy simple, primero inicializamos el valor de retorno al id de la operación, pues si la operación no se cierra se devuelve este mismo valor, si se cierra será -1 y en otra variable booleana guardamos el valor devuelto por la función predefinida OrderClose, de la cual sabemos su funcionamiento gracias a la documentación correspondiente de mql4 [[10]](#bibliografía). La función recibirá el identificador de la operación que se quiere cerrar (id), el lotaje, el precio de salida (Bid si es de compra y Ask si es de venta) y por último el slipagge, que será de 0 para nuestras operaciones.

Por último, se comprueba si la operación se ha cerrado, en ese caso se retornará un -1, en caso contrario se mantiene el identificador propio de la operación, pues continuaría abierta.

//Comprueba si existe alguna operación abierta con

//el magic number proporcionado

//Parámetros: idmagic: magic number de operación

//Retorno: Si no existe ninguna operación con el magic dado devuelve -1

//si existe alguna operación devuelve el identificador de la operación

int OperacionAbierta(int idmagic){

int total = OrdersTotal(); //Ordenes totales abiertas

int r = -1;//valor de retorno de la función

La función OperacionAbierta será la que inicializará los valores de los identificadores de idcorta e idlarga al iniciar el sistema, buscando si hay alguna operación abierta de este tipo a partir de su magicnumber. Creamos dos variables enteras, una que contendrá el número total de operaciones en el sistema en el momento de la comprobación, este valor se obtiene mediante la función predefinida OrdersTotal (). La otra será el valor de retorno al comprobar si hay operación abierta o no, que por defecto es -1, pues se supone que al iniciar el sistema no se han abierto operaciones aún.

//se busca entre todas las operaciones abiertas en el sistema

//si alguna de ellas coincide con el magic number dado(idmagic)

//si se encuentra una operación con ese magic el valor de retorno

//de la función(r) será el identificador de la operación

//y paramos de buscar

//si no se encuentra el valor de retorno(r) es -1

//que significa que no hay operaciones abiertas y

//por tanto podremos abrir una

for(int i = 0 ; i < total && r == -1 ; i++){

//true si encuentra esa operación, false si no la encuentra

bool exito = OrderSelect(i , SELECT\_BY\_POS , MODE\_TRADES);

//si encuentra la operación y el maginumber

//de esta operación es igual al dado

if(exito && OrderMagicNumber() == idmagic){

//Si la fecha de cerrado es 0 es que hay operación abierta

if(OrderCloseTime() == 0) r = OrderTicket();

}

}

return r;

}//Fin OperacionAbierta

Recorremos todas las operaciones abiertas mientras no se encuentre nuestra operación. Lo haremos de la siguiente manera:

1. Intentamos seleccionar la operación mediante la función predefinida OrderSelect.
2. En caso de encontrar una operación abierta, comprobamos si el magicnumber de esa operación (OrderMagicNumber () nos lo da) coincide con el magicnumber de la operación que buscamos, si no, seguimos buscando en las siguientes operaciones.
3. Si los magicnumbers coinciden, comprobamos la fecha de cerrado de la operación con OrderCloseTime (), que deberá ser 0 si la operación está abierta.

Si se cumplen todos los pasos mencionados anteriormente, la función retornará el valor del identificador de la operación en cuestión, en caso contraria devolverá -1.

Esta función se ha diseñado basándonos en los conocimientos adquiridos en el curso impartido por el tutor [[3]](#bibliografía) y en la documentación de mql4 para el magicnumber [[11] y el OrderSelect [12]](#bibliografía).

**4 - EXPERIMENTOS Y VA****LIDACIÓN**

Ahora que ya tenemos programado el sistema, conocemos los aspectos importantes de su implementación y lo hemos probado de forma manual en distintos mercados y timeframes para comprobar que el funcionamiento es el esperado, llega la fase más importante del desarrollo del sistema, la optimización.

Para llevar a cabo la fase de optimización, realizaremos los pasos explicados en el apartado de cómo hacer la optimización y su importancia, pero aplicados a un mercado en concreto, que será en el que hemos encontrado el set rentable y no sobreoptimizado.

**4.1 - ELECCIÓN DE MERCADO Y TIMEFRAME**

Antes de empezar con la optimización en sí, el primer paso es escoger en que mercado vamos a probar el sistema, aunque el RSI y el indicador de Ichimoku son universales y, por tanto, podrían utilizarse para cualquier mercado, debido a su comportamiento hay algunos mercados a los que se adaptarán mejor que a otros.

El RSI e Ichimoku son indicadores útiles para detectar posibles cambios de tendencia, por lo que deberíamos empezar a probar por mercados tendenciales. Se ha decidido empezar a probar por el mercado de las materias, en concreto el oro, ya que su precio puede ser influenciado por factores externos, como la inflación o la política monetaria y es un mercado al que nuestro sistema se adaptaría bien.

Aunque solo se expondrá el caso del oro en este documento, el sistema puede ser probado también en otro tipo de mercados tendenciales, como podría ser el S&P500.

Ahora que ya tenemos escogido el mercado, solo nos queda escoger el timeframe, lo ideal, es que empecemos por el timeframe más alto posible, ya que a menor sea el timeframe mayor aleatoriedad en el comportamiento del sistema y, por tanto, más difícil encontrar un set rentable no sobreoptimizado. Para nuestro caso utilizaremos el timeframe de 4 horas.

**4.2 - CONFIGURACIÓN DE PARÁMETROS**

Una vez hemos encontrado un mercado para probar y hemos seleccionado un timeframe, debemos configurar los rangos de los parámetros de nuestro sistema.

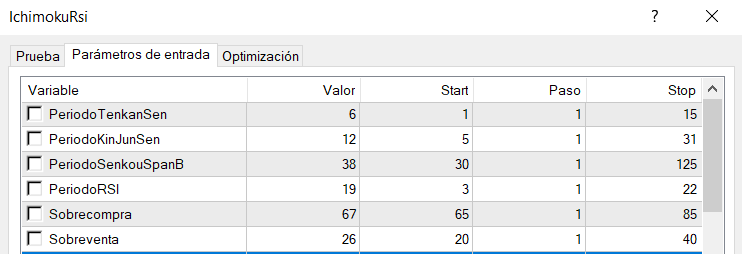


Ilustración 4: Configuración de los rangos de valores para los parámetros optimizables del sistema

La ilustración 4 muestra el rango de variación de los parámetros del sistema. La columna start indica el valor inicial (límite inferior del rango) y la columna Stop el valor final (límite superior del rango). Solo se muestran los parámetros optimizables del sistema, pues los otros parámetros son el magicnumber (identificador de operación) y el lotaje, el cual no tiene mucho sentido optimizar, pues a mayor lotaje, mayor beneficio o perdida y a menor lotaje menor beneficio o perdida.

A continuación, se explica el porqué del rango de valores escogido para cada parámetro.

Para los periodos referentes al indicador de Ichimoku (PeriodoTenkanSen, PeriodoKinjunSen y PeriodoSenkouSpanB), nos hemos basado en la información obtenida en un artículo de la página Forex.ins.rs [[13],](#bibliografía) que nos dice que para los mercados que solo se abren 5 días a la semana y los mercados tendenciales, hay tres posibles combinaciones de parámetros además de la configuración por defecto de Ichimoku:

* 9, 26, 52 (configuración por defecto)
* 9, 30, 60
* 7, 22, 44
* 12, 24, 120

El primer valor es el periodo TenkanSen. El segundo es el periodo KijunSen y el tercero es el periodo para el cálculo del Senkou Span B. Para escoger el rango de variación, nos fijamos en las cuatro configuraciones expuestas anteriormente y tomamos el valor más grande y más pequeño para cada parámetro.

* TenkanSen: su valor más pequeño es el 7 y el mayor el 12, por tanto, nos quedamos con un rango de 7 a 12
* KijunSen: su valor más pequeño es el 22 y el más grande el 30, por tanto, nos quedamos con un rango de 22 a 30
* Senkou Span B: su valor más pequeño es el 44 y su valor más grande es el 120, por tanto, nos quedamos con un rango de 44 a 120

Estos rangos hacen referencia al rango de valores válidos para estos parámetros, por tanto, sería adecuado aumentar este límite tanto por arriba como por abajo, pues en la optimización, podría pasar que estos valores se encontrarán en bordes del gráfico y no centros de mesetas y los descartásemos por esta causa cuando podrían ser buenos valores. Por tanto, el rango de valores para estos parámetros sería el siguiente:

* TenkanSen: como anterior rango teníamos del 7 al 12, añadimos valores por arriba y por abajo, ahora el nuevo rango es del 1 al 15
* KijunSen: como anterior rango teníamos del 22 al 30, añadimos valores por encima y por abajo, ahora el nuevo rango es del 5 al 31
* Senkou Span B: como anterior rango teníamos del 44 al 120, añadimos valores por encima y por debajo, ahora el nuevo rango es del 30 al 125

Con estos rangos nos aseguramos que los valores válidos de los parámetros no se encuentran en bordes del gráfico y además aumentamos las posibilidades de encontrar un set rentable, pues probamos más valores.

Para elegir el rango de variación del periodo del RSI, nos hemos basado en el siguiente texto:

” Siéntete libre de probar el RSI en diferentes mercados con diferentes períodos. Un RSI 14 puede funcionar bien en DAX 30, por ejemplo, pero funciona mejor con un parámetro de 21 en criptomonedas u oro. Siéntete libre de probar los ajustes correctos en el mercado que más te convenga” [[Fuente: 1],](#bibliografía) el cual se aplica a la filosofía de nuestro sistema, ya que como hemos mencionado, es un sistema que se adapta bien a mercados como el oro. También nos hemos basado en el siguiente texto: “El período N que se considera habitualmente para el cálculo del RSI es igual a 14” [[Fuente: 2].](#bibliografía)

Por tanto, a partir de la información citada en el párrafo anterior, tenemos que el periodo más habitual para el RSI es el 14 y el que mejor se adapta a mercados como el oro el 21, que son los mercados que nos interesan.

Al igual que para elegir los rangos de variación de los parámetros de Ichimoku, aumentamos los límites para no dejar ningún valor en un borde y explorar nuevas posibilidades.

El rango de valores para el RSI, quedaría de la siguiente manera:

* PeriodoRSI: el valor más habitual del RSI es el 14 y para el oro, que es el mercado que nos interesa el 21, por tanto, el rango de valores se establece del 3 al 22, aumentando así los límites de valores explorables.

Por último, solo nos queda por decidir el rango de variación para los valores de sobrecompra y sobreventa, para estos rangos nos hemos basado en la información dada en uno de los videos del curso “Trading algorítmico desde cero con Metatrader 4 y MQL4” [[3]](#bibliografía), perteneciente en este caso al Tutor de este trabajo.

De ese video obtenemos que los valores por encima del valor 50, son tendencia alcista y por debajo de 50 son tendencia bajista, además de los límites de sobrecompra y sobreventa, donde un valor por debajo de 30, indicaría zona de sobreventa y un valor por encima de 70, indicaría zona de sobrecompra.

Además de la información sacada del video se ha extraído información también de la siguiente tabla: [[Fuente: 1]](#bibliografía)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Señal rápida | Señal normal | Señal lenta |
| El mercado se encuentra en una zona de sobrecompra | 70 | 80 | 90 |
| El mercado se encuentra en una zona de sobreventa | 30 | 20 | 10 |

Y el siguiente texto: “Cuanto más cerca se encuentre el nivel de la zona de los 100, menos señales de trading se identificarán con el indicador RSI. Cuanto más cerca esté el nivel de 50, más señales tendrá (incluyendo señales falsas).” [[Fuente: 1]](#bibliografía)

Por tanto, tomando en cuenta las distintas fuentes de información, los rangos establecidos para la sobrecompra y sobreventa del RSI son los siguientes:

* Sobrecompra: entre 65 y 85, basándonos en los valores de señal rápida y lenta de la tabla anterior (70, 80) y aumentando el límite para no dejar estos valores en los bordes
* Sobreventa: entre 20 y 40, abarcando los valores de señal rápida y lenta (20, 30) y aumentando el límite por arriba para incluir más valores de la zona bajista (por debajo de 50)

**4.3 – PERIODO DE PRUEBA Y CONFIGURACIONES PREVIAS**

Una vez que tenemos decidido el mercado y los rangos de variación de nuestros parámetros, es hora de elegir el periodo de prueba. El periodo escogido debe ser de al menos 10 años, para este caso se ha elegido el periodo comprendido entre el 01/01/2013 – 01/05/2023. Este periodo de prueba se divide en dos partes: Una primera parte que comprenderá los dos primeros tercios del histórico (01/01/2013 – 01/01/2020) y otra que comprenderá el último tercio del histórico (01/01/2020 – 01/05/2023).

En la primera parte del histórico realizaremos la optimización de los parámetros del sistema. La segunda la emplearemos para hacer el backtest final, que nos dirá si nuestro set es válido o no.

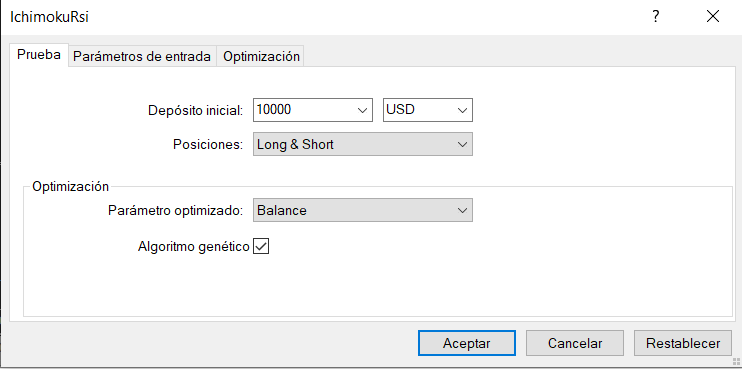


Ilustración 5: Balance inicial, función objetivo y marcado de algoritmo genético

En la pestaña de propiedades del experto del probador de estrategias, seleccionamos el apartado de prueba (ilustración 5) y marcamos la opción de Algoritmo genético, por el cual la optimización será genética, si no se marca, se hará una optimización por fuerza bruta.

Seleccionamos el balance inicial, que será de 10000, y en parámetro optimizado seleccionamos balance, el parámetro optimizado no es más que la función objetivo, que se describió en la sección de introducción.

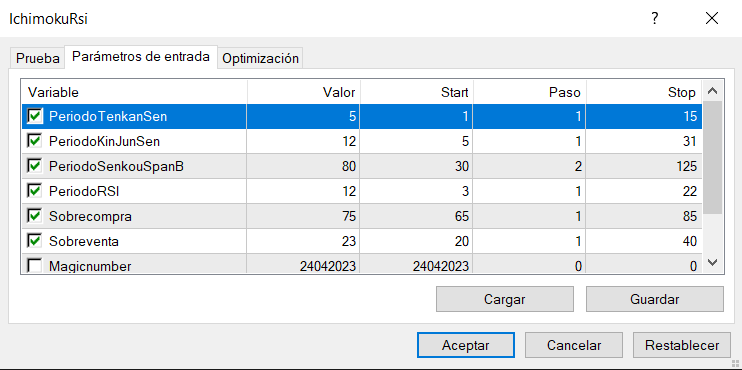


Ilustración 6: Marcado de parámetros del sistema

Ahora nos vamos a la parte de parámetros de entrada y marcamos todos los parámetros que vayamos a optimizar del sistema (ilustración 6).

**4.4 - OPTIMIZACIÓN GENÉTICA**

Una vez hemos elegido un mercado para probar, un timeframe, hemos configurado y establecido el rango de valores para los parámetros de nuestro sistema, ya podemos empezar con el proceso de optimización.

El primer paso de la optimización, será realizar una optimización genética de todos los parámetros de nuestro sistema a la vez en el primer tercio de nuestro histórico(01/01/2013 – 01/01/2020), de aquí obtendremos nuestros primeros sets candidatos. Serán seleccionados como sets candidatos en este parte, aquellos sets que generen beneficios, realicen más de 50 operaciones y tengan unos valores distintos entre sus parámetros.

Estos sets serán nuestro punto de partida en el proceso de optimización. Puede que, al terminar de optimizar, el set final válido no tenga nada que ver con ninguno de estos.

A continuación, se muestran algunos ejemplos del tipo de set que nos interesa (se ha probado con más, pero los que se muestran nos valen para entender el proceso de selección):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 5 |  | 6 | 46 | 6 | 80 | 27 |
| 2º | 4 |  | 25 | 36 | 20 | 70 | 32 |
| 3º | 5 |  | 8 | 66 | 14 | 77 | 21 |
| 4º | 14 |  | 21 | 120 | 3 | 71 | 36 |

De los cuatro sets mostrados, continuaremos el proceso de optimización con el primero.

**4.5 - PRIMERA OPTIMIZACIÓN POR FUERZA BRUTA**

Una vez seleccionado uno o varios de los sets obtenidos en la optimización genética, ahora debemos optimizar por fuerza bruta los dos parámetros más importantes del sistema y que están correlacionados entre sí, en nuestro caso, esta primera optimización la realizaremos para los periodos de las líneas rápida y lenta (Tenkan-Sen y Kijun-Sen). El periodo para esta optimización será el mismo que para la optimización genética, los dos primeros tercios del histórico.

Para esta parte no bastará con seleccionar los sets que más beneficios den, estos sets deben ser el centro de mesetas en el gráfico resultante. Recordemos, una meseta no es más que una zona del gráfico donde los beneficios cambian de manera suave y progresiva, esto se verá en el gráfico porque será una zona donde la intensidad de color se mantendrá similar.

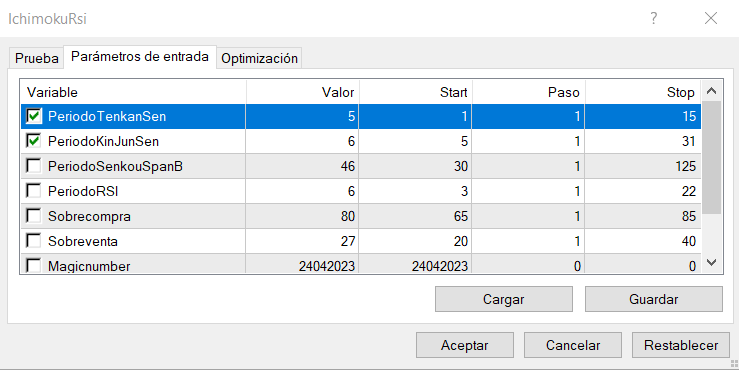


Ilustración 7: Configuración de parámetros para la primera optimización por fuerza bruta

Para realizar la optimización por fuerza bruta, debemos quitar la opción de algoritmo genético en el apartado de ajustes y seleccionar únicamente los parámetros a optimizar, dejando el resto igual, como se puede ver en la ilustración 7.

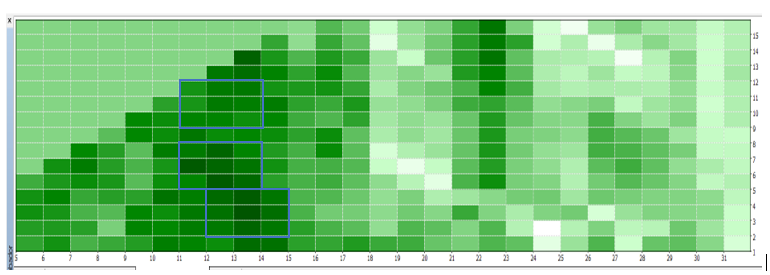


Ilustración 8: Primera optimización por fuerza bruta correspondiente al periodo comprendido entre el 01/01/2013 – 01/01/2020 en el mercado del oro, con un timeframe de 4 horas. El eje x representa el periodo Kijun-Sen, el eje y representa el periodo Tenkan-Sen. En azul se resaltan las mesetas encontradas

En la ilustración 8, se muestra el gráfico resultante de la primera optimización por fuerza bruta, con las mesetas encontradas remarcadas en azul. De estas mesetas, nos quedaremos con sus valores centrales.

Por tanto, los sets escogidos de este primer paso serán los siguientes:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 10 |  | 12 | 46 | 6 | 80 | 27 |
| 2º | 3 |  | 13 | 46 | 6 | 80 | 27 |
| 3º | 6 |  | 12 | 46 | 6 | 80 | 27 |

En concreto, vamos a continuar la optimización con el tercer set.

**4.6 - SEGUNDA OPTIMIZACIÓN POR FUERZA BRUTA**

Una vez optimizados los dos primeros parámetros del sistema, debemos continuar optimizando los dos siguientes parámetros, en nuestro caso serán el PeriodoSenkouSpanB y el PeriodoRSI. Para ello, partiremos del tercer set válido obtenido en la primera optimización por fuerza bruta.

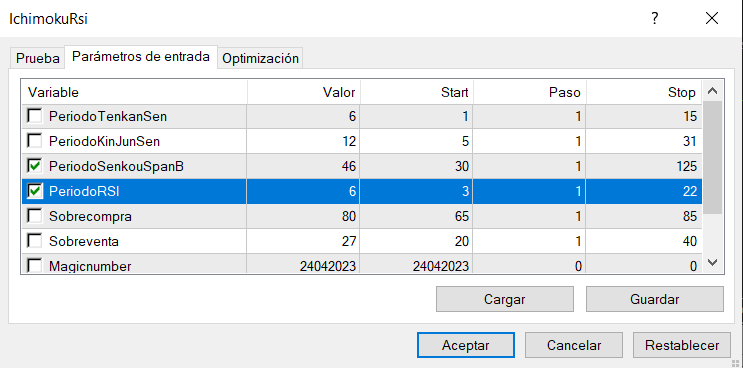


Ilustración 9: Configuración de parámetros, para la segunda optimización por fuerza bruta

Ahora marcamos los dos parámetros a optimizar y dejamos igual el resto, como se puede ver en la ilustración 9.

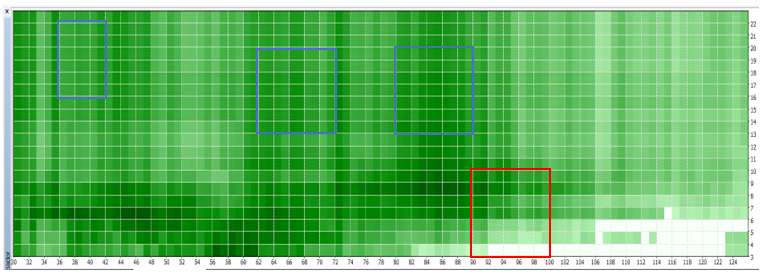


Ilustración 10: Segunda optimización por fuerza bruta correspondiente al periodo comprendido entre el 01/01/2013 – 01/01/2020 en el mercado del oro, con un timeframe de 4 horas. El eje x representa el Periodo de SenkouSpan B, el eje y, representa el periodo para el cálculo del RSI. En azul se resaltan las mesetas encontradas, en rojo se resalta un ejemplo de meseta no válida

La ilustración 10, muestra el resultado de la segunda optimización por fuerza bruta, en este caso, ante la gran cantidad de mesetas que contiene el gráfico, solo se marcan algunas de ellas (remarcadas en azul en la ilustración 10). Al igual que antes, tomaremos los valores centrales de estas mesetas, dando lugar a los siguientes sets:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 67 | 16 | 80 | 27 |
| 2º | 6 |  | 12 | 85 | 16 | 80 | 27 |
| 3º | 6 |  | 12 | 38 | 19 | 80 | 27 |

De estos sets, vamos a continuar la optimización con el primero y el tercero. Antes de continuar con el proceso de optimización, se ha remarcado también como ejemplo (recuadro rojo de la ilustración 10) un set que no sería rentable. Este set no es rentable, pues como se puede apreciar en la ilustración 10, no cumple con las características necesarias para ser un set válido, pues no es una meseta, presenta cambios de valor muy bruscos. El set en cuestión sería el siguiente:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 94 | 6 | 80 | 27 |

A continuación, para completar la información obtenida del gráfico, se muestra la evolución de los beneficios de los sets rentables y del no rentable a medida que nos movemos por el eje y (PeriodoRSI).

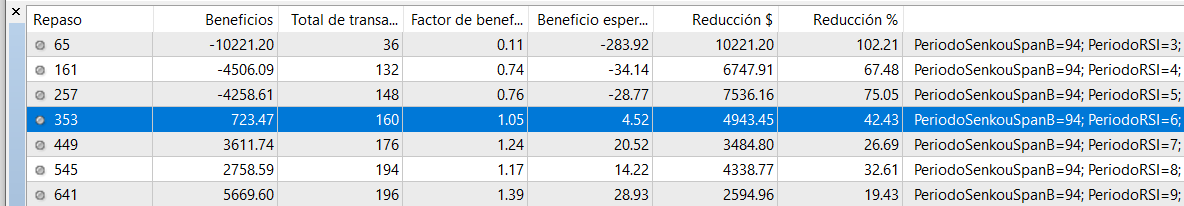


Ilustración 11: Evolución de beneficios del set no válido escogido a medida que vamos moviéndonos por el eje y del gráfico (periodo RSI)

La ilustración 11, nos confirma lo que se veía en el gráfico, este set está sobreoptimizado, pues no puede ser que con valor 6 de periodo del RSI nos de beneficios, pero con valor 5 cambiando solo un punto, nos dé perdidas, por tanto, este set está sobreoptimizado.

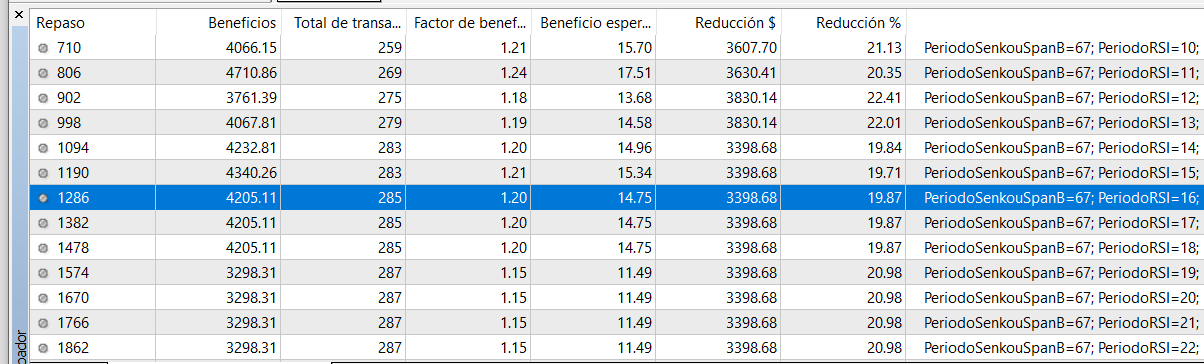


Ilustración 12: Evolución de beneficios del primer set válido escogido a medida que vamos moviéndonos por el eje y del gráfico (periodo RSI)

La ilustración 12, refleja la evolución de los beneficios del primer set escogido como válido, como se puede ver el cambio es mucho más gradual, a medida que vamos cambiando el valor del periodo del RSI los beneficios cambian, pero de una forma mucho más suave que el anterior set (ilustración 11). Por tanto, esto nos reafirma lo que habíamos visto en el gráfico, que este es un buen set candidato.

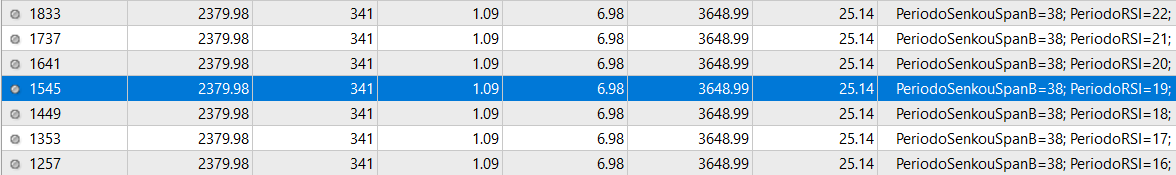


Ilustración 13: Evolución de beneficios del segundo set válido escogido a medida que vamos moviéndonos por el eje y del gráfico (periodo RSI)

Como se puede apreciar en la ilustración 13, la evolución de los beneficios del segundo set escogido como válido, también es estable (en este caso su valor no cambia), por tanto, nos confirma lo que ya habíamos visto en el gráfico, que este set es bueno.

**4.7 - TERCERA OPTIMIZACIÓN POR FUERZA BRUTA**

Ya tenemos optimizados cuatro de nuestros seis parámetros, por tanto, ahora solo nos falta por optimizar los dos últimos: Sobrecompra y Sobreventa. El procedimiento en este caso es el mismo que en el anterior caso, partimos de los sets válidos seleccionados en la anterior optimización.

A continuación, a modo de recordatorio se exponen estos sets:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 67 | 16 | 80 | 27 |
| 2º | 6 |  | 12 | 38 | 19 | 80 | 27 |

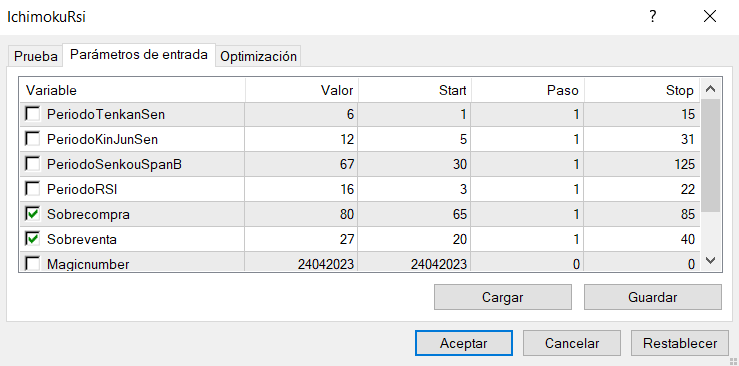


Ilustración 14: Configuración de parámetros para la tercera optimización por fuerza bruta. Esta configuración pertenece al primer set escogido como válido.

Como se puede ver en la ilustración 14, en este caso marcamos la sobrecompra y sobreventa y mantenemos el resto de parámetros igual.

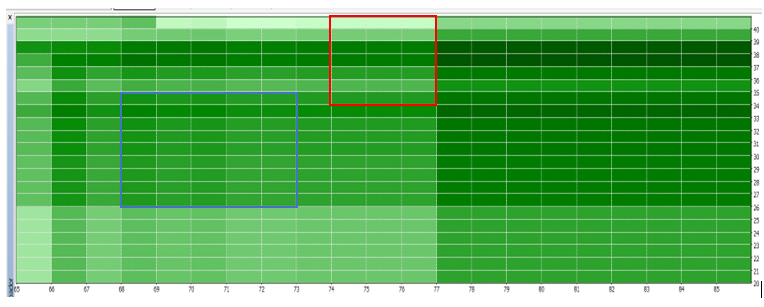


Ilustración 15: Tercera optimización por fuerza bruta del primer set escogido, correspondiente al periodo comprendido entre el 01/01/2013 – 01/01/2020 en el mercado del oro, con un timeframe de 4 horas. El eje x representa el valor de sobrecompra, el eje y representa el valor de sobreventa. En azul se resalta la meseta encontrada, en rojo se resalta un ejemplo de meseta no válida

En la ilustración 15, se muestra el resultado de la tercera optimización por fuerza bruta del primer set escogido en el paso anterior. En este caso, el set seleccionado como válido sería el siguiente:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 67 | 16 | 70 | 30 |

Al igual que en la segunda optimización, se remarca en rojo, un ejemplo de set no válido y que, por tanto, rechazaríamos en este paso. El set sería el siguiente:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 67 | 16 | 75 | 37 |

Una vez hemos probado el primero de nuestros sets, debemos probar ahora el segundo de ellos.

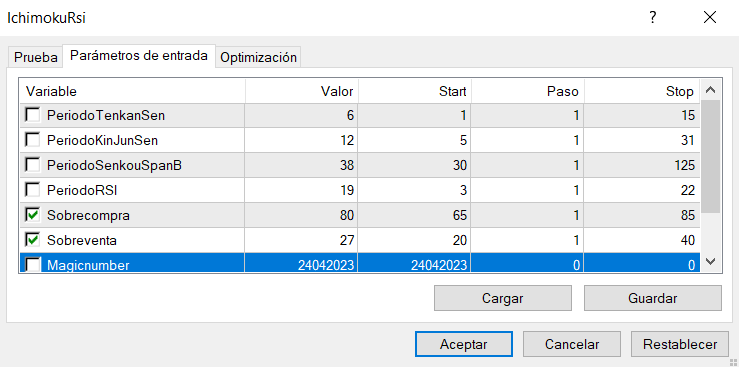


Ilustración 16: Configuración de parámetros para la tercera optimización por fuerza bruta. Esta configuración pertenece al segundo set escogido como válido.

Antes de optimizar debemos cambiar los valores de los parámetros por los correspondientes al segundo set. En la ilustración 16, se muestra la configuración para probar el otro set rentable que habíamos obtenido, en este caso, cambiamos los valores del PeriodoSenkouSpanB y el PeriodoRSI a los valores que le corresponden.

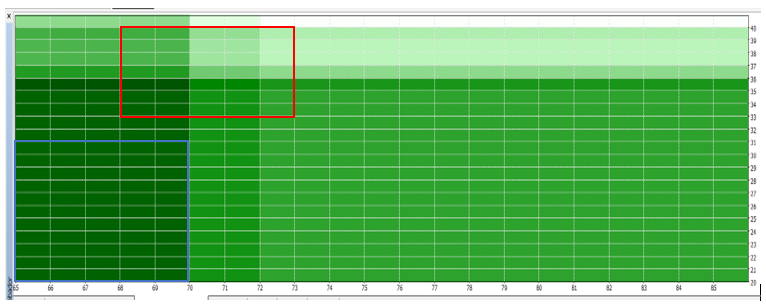


Ilustración 17: Tercera optimización por fuerza bruta del segundo set válido escogido, correspondiente al periodo comprendido entre el 01/01/2013 – 01/01/2020 en el mercado del oro, con un timeframe de 4 horas. El eje x representa el valor de sobrecompra, el eje y representa el valor de sobreventa. En azul se resalta la meseta encontrada, en rojo se resalta un ejemplo de meseta no válida

La ilustración 17, muestra el resultado de la optimización por fuerza bruta de los valores de sobrecompra y sobreventa, para el segundo set escogido. El set válido en este caso sería el siguiente:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 38 | 19 | 67 | 25 |

El set no válido sería el siguiente en este caso:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 38 | 19 | 70 | 36 |

Por tanto, los sets que nos quedaríamos como válidos tras la tercera optimización por fuerza bruta, serían los siguientes:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 67 | 16 | 70 | 30 |
| 2º | 6 |  | 12 | 38 | 19 | 67 | 25 |

Y los no válidos:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 67 | 16 | 75 | 37 |
| 2º | 6 |  | 12 | 38 | 19 | 70 | 36 |

A continuación, al igual que en la anterior optimización, se muestra la evolución de los beneficios de los sets escogidos en este paso.

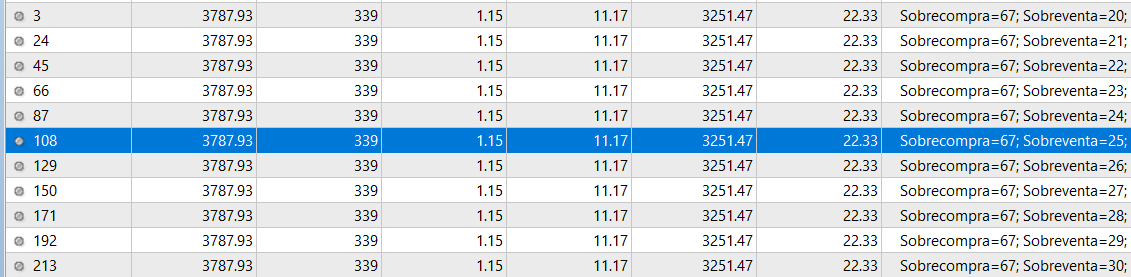


Ilustración 18: Evolución de los beneficios del segundo set escogido como válido

En la ilustración 18, se puede ver como a medida que cambiamos el valor de la sobreventa, los beneficios no cambian, lo que confirma que este set es válido.

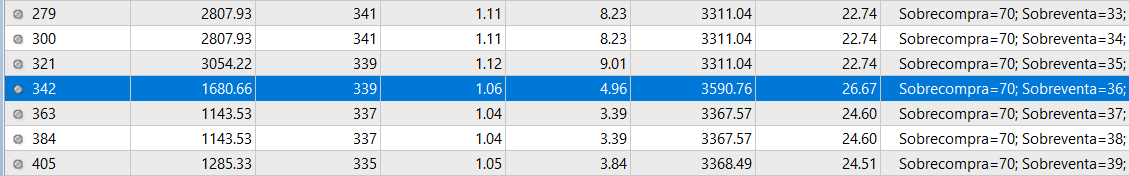


Ilustración 19: Evolución de los beneficios del segundo set escogido como no válido

En la ilustración 19, se puede ver como la evolución de los beneficios para este set es mucho más brusca que en el anterior, simplemente cambiando el valor de sobreventa de 36 a 35, los beneficios casi se duplican. Esto nos confirma lo que ya intuíamos por el gráfico, que este set no es bueno.

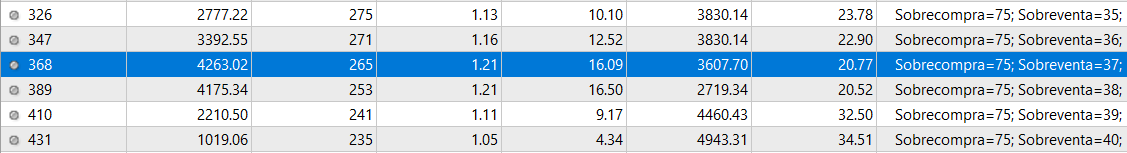


Ilustración 20: Evolución de los beneficios del primer set escogido como no válido

La ilustración 20 nos confirma lo que habíamos visto en el gráfico, que el set posiblemente este sobreoptimizado, ya que, aunque no llegue a dar perdidas, en cuanto pasamos del valor 37 de sobreventa al valor 39, sus beneficios se reducen a la mitad.

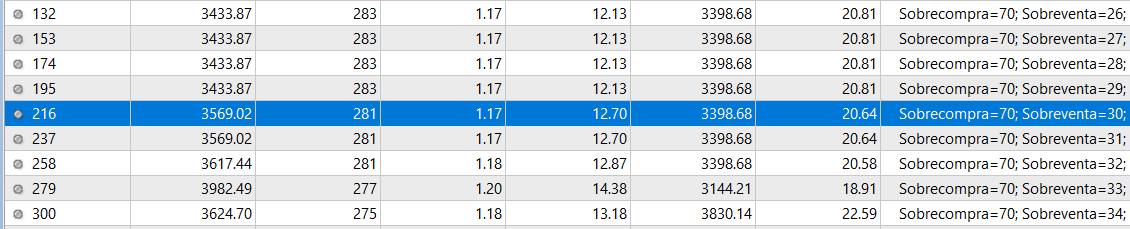


Ilustración 21: Evolución de beneficios del primer set escogido como válido

La ilustración 21 nos confirma que el set que hemos escogido como válido realmente lo es, a medida que vamos bajando y subiendo los valores de sobreventa los beneficios apenas cambian.

**4.8 - BACKTEST FINAL**

Para concluir con el proceso de optimización, es necesario probar los posibles sets que no estén sobreoptimizados encontrados durante la optimización. Para hacer esto, realizaremos una prueba retrospectiva (backtest), con cada uno de estos sets en el último tercio de nuestro histórico, que en este caso corresponde al período desde el 01/01/2020 hasta el 01/05/2023.

Si el set que probamos supera esta última prueba, tendríamos un set rentable y no sobreoptimizado que guardaríamos. Para que un set sea válido después de esta fase de pruebas, debe cumplir dos condiciones: primero, el número de operaciones en este período debe ser superior a 50. Si no es así, a pesar de que el set sea rentable, tendríamos que descartarlo por no ser estadísticamente relevante. En segundo lugar, en caso de que el set tenga más de 50 operaciones, debemos fijarnos en su gráfica de balance: debe obtener beneficios y la evolución del balance debe ser lo más recta posible.

Antes de continuar, recordemos los sets que se va a probar en este proceso:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Nº set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 67 | 16 | 70 | 30 |
| 2º | 6 |  | 12 | 38 | 19 | 67 | 25 |

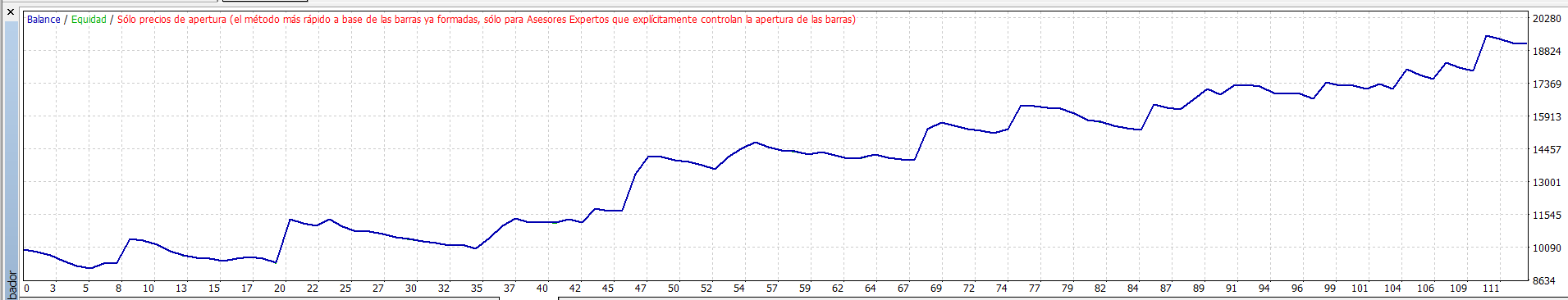


Ilustración 22: Gráfico de balance final del primer set válido escogido en la última fase de optimización por fuerza bruta, correspondiente al periodo comprendido entre 01/01/2020 – 01/05/2023 para el mercado del oro, con un timeframe de 4 horas. El eje x hace referencia al número de operación, el eje y al balance total

Como se puede ver la ilustración 22 (referente al primer set válido mencionado al final de la página anterior), la evolución del balance en el primer tercio no es del todo buena, pues presenta altibajos, pero en el segundo tercio si es muy buena, pues asciende de manera constante. Además, se realizan más de 50 operaciones en el periodo final, por tanto, también cumple esta condición. Al cumplir las dos condiciones mencionadas anteriormente (balance creciente y más de 50 operaciones), podemos afirmar que el set escogido como válido en el paso anterior realmente lo es.

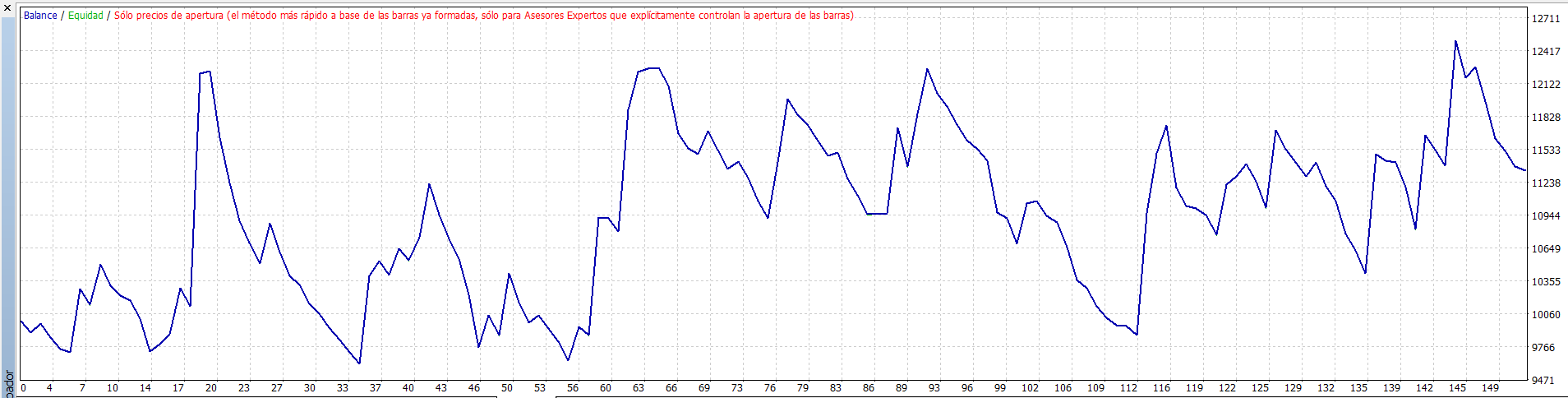


Ilustración 23: Gráfico de balance final del segundo set válido escogido en la última fase de optimización por fuerza bruta, correspondiente al periodo comprendido entre 01/01/2020 – 01/05/2023 para el mercado del oro, con un timeframe de 4 horas. El eje x hace referencia al número de operación, el eje y al balance total

En este caso, como se puede ver en la ilustración 23 (referente al segundo set válido mencionado al final de la página anterior) el set da beneficios y realiza más de 50 operaciones, pero su evolución no es constante, tiene muchos altibajos, por tanto, este set lo descartaríamos.

Por tanto, una vez visto el comportamiento de los dos sets en el tramo final, nos vamos a quedar como set válido el primero de los dos. Este set válido es el siguiente:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 67 | 16 | 70 | 30 |

**4.9 - INFORME FINAL**

Ya hemos visto que nuestro set genera beneficios en el último tercio del histórico y que, además, es un set relevante, porque hace más de 50 operaciones y su gráfica es ascendente durante todo el periodo, para confirmar que este set es bueno realmente y que hemos hecho bien el proceso, también es importante tener en cuenta otras métricas, que se explican a continuación.

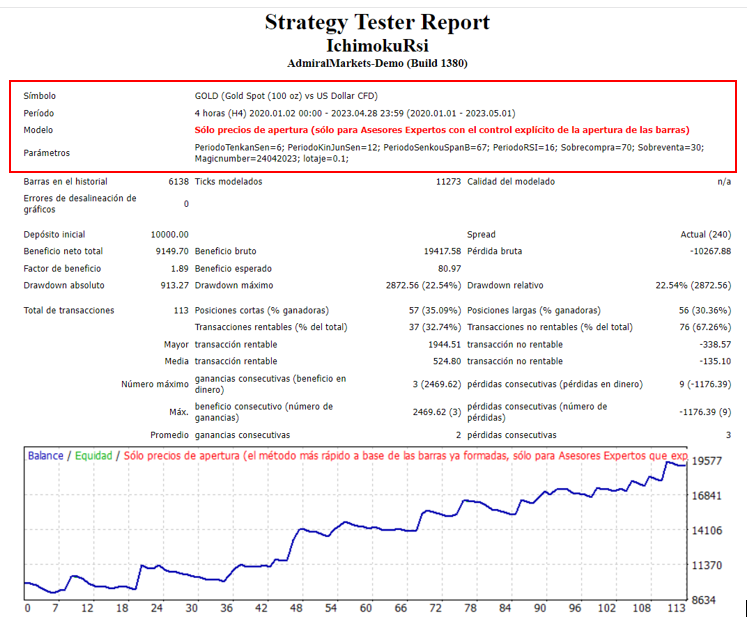


Ilustración 24: Informe final del set válido escogido, tras la última optimización por fuerza bruta del sistema, remarcado en rojo, se encuentra: símbolo (mercado probado), Periodo (timeframe e intervalo de prueba), Modelo (apertura de velas) y los parámetros del set.

El primer concepto importante y que debemos tener en cuenta, es el drawdown máximo, que según un artículo de admiral markets:” El máximo drawdown será la diferencia entre el máximo más alto de la cartera hasta el mínimo más bajo posterior” [[Fuente: 4]](#bibliografía), a mayor sea este valor, más riesgo tendrá el set encontrado, por tanto, puede ser un factor importante a tener en cuenta a la hora de escoger un set, ya que tener un control de riesgos también es parte de una buena estrategia.

Además, teniendo en cuenta las siguientes afirmaciones de distintas fuentes de información:” Sin embargo, siempre se recomienda a los inversores y a los traders que el drawdown se mantenga por debajo del nivel del 20%.” [[Fuente: 6],](#bibliografía)

“Pero tampoco recomiendo operar con estrategias o asesores expertos que tengan niveles de Drawdown Máximo superiores al 25%” [[Fuente: 5]](#bibliografía).

Nuestro set presenta un drawdown máximo del 22,54%, encontrándose entre medias de lo mencionada en las anteriores fuentes, por tanto, nuestro drawdown sería medio, encontrándose dentro de unos límites de riesgo asumibles.

Otro concepto a tener en cuenta es el drawdown absoluto, que según un artículo de earnforex sería lo siguiente: “El Drawdown Absoluto es la diferencia entre el depósito inicial y el punto mínimo por debajo de nivel del depósito durante todo el período de revisión. Le dice cómo de grande podría ser su pérdida comparada con el depósito inicial durante el trading.” [[Fuente: 5]](#bibliografía)

En nuestro caso es de 913.27, que no supone un riesgo excesivamente alto (menos del 10% del depósito), ya que partíamos de un capital inicial mucho más alto (10000).

El drawdown relativo, según un artículo de la página fbs, se define como: “El drawdown relativo (reducción relativa) es la caída máxima de tu capital en términos de porcentaje. Se puede calcular como el drawdown máximo dividido por el valor de capital máximo multiplicado por 100%”. [[Fuente: 7].](#bibliografía) En nuestro caso es de 22,54%, lo que indica un riesgo medio.

Los tres tipos de drawdowns son métricas que debemos tener en cuenta a la hora de considerar usar un set o no, ya que al final nos están definiendo el riesgo que va a tener el sistema, a partir de ahí, ya decidiremos nosotros si estamos dispuestos a asumirlo.

Otro valor a tener en cuenta es el porcentaje de operaciones ganadoras y perdedoras, en este caso de las 113 operaciones realizadas en el periodo 37 han sido ganadoras y 76 perdedoras, lo que da un 32.74% de operaciones ganadoras, es decir que aproximadamente 3 de cada 10 operaciones son ganadoras y 7 de 10 perdedoras. Esto, en un principio, no es un buen indicativo, pues nuestro sistema estaría haciendo más operaciones perdedoras que ganadoras, pero si nos fijamos en la media de transacción rentable(media de ganancias en cada operación ganadora) y en la media de transacción no rentable(media de perdidas en cada operación perdedora), tenemos que las operaciones rentables de media obtienen 524.80 de beneficio, frente a los 135.10 que pierden las perdedoras, esto supone que de media por cada operación que ganamos, recuperamos más de 4 veces la perdida, lo que compensa el tener un mayor número de operaciones perdedoras. Además, si nos fijamos en la mayor operación rentable obtenida, tenemos que es de 1944.51, frente a 338.57 de la mayor operación no rentable. Es decir, que la peor operación del periodo probado se encuentra por debajo de la media de beneficios por operación (524.80 frente a 338.57), por lo que realmente, el riesgo de pérdida es bajo.

También, debemos tener en cuenta el beneficio esperado, que nos es más que coeficiente resultante de dividir las ganancias de las operaciones ganadoras entre las pérdidas de las operaciones perdedoras. Lo ideal, es que este valor sea lo más alto posible y siempre superior a 1. En este caso, el factor de beneficio es de 1.89, lo que significa que por cada dólar que perdemos ganamos 1.89, lo cual es un factor de beneficio bastante bueno.

Con los datos obtenidos del set en el último tercio y teniendo en cuenta lo explicado hasta ahora, podemos afirmar que este set no está sobreoptimizado. Esto no significa que podamos poner el set a operar en una cuenta real, el siguiente paso tras las pruebas sería poner el set a operar en cuenta demo al menos 3 meses, para ver cómo se comporta en mercado real y ahí ya decidir si lo utilizamos o no.

**5 - CONCLUSIONES**

A lo largo del documento, se ha evidenciado que el desarrollo de un sistema de trading algorítmico rentable es una tarea complicada que requiere de minuciosidad en cada paso, no solo durante las fases de diseño y optimización, sino desde los primeros pasos. Es fundamental seleccionar los indicadores adecuados y combinarlos de manera correcta para aumentar la probabilidad de encontrar un set rentable. Si realizamos esta selección de forma arbitraria, es muy probable que no encontremos ningún set rentable y habremos desperdiciado nuestro tiempo e incluso dinero si llegásemos a operar condiciones reales.

En nuestro caso, hemos decididio utilizar los indicadores de Ichimoku y el RSI. Entre estos dos indicadores, el más relevante es el de Ichimoku, ya que nos proporcionará información sobre las tendencias del mercado. Cuando el precio cruza el límite superior de la nube, indica que la tendencia existente (lateral o bajista) está llegando a su fin y nos acercamos a una tendencia alcista. Por otro lado, cuando la nube cruza hacia abajo el límite, indica el final de una tendencia alcista o lateral y el comienzo de una tendencia bajista.

Aunque el indicador Ichimoku ofrece un análisis completo del mercado, es recomendable complementarlo con otro indicador para obtener mayor seguridad. En nuestro caso, utilizamos el RSI. La función del RSI en el sistema es confirmar las tendencias señaladas por Ichimoku. Si se produce un cruce del precio con el límite superior de la nube, el valor del RSI debe encontrarse por debajo del nivel de sobrecompra. Esto se debe a que cuando nos encontramos en sobrecompra, significa que el activo está siendo comprado en exceso, lo que indica altas probabilidades de un cambio de tendencia. Por lo tanto, es mejor evitar operar en estas zonas.

Para las tendencias bajistas, ocurre algo similar. Si el precio cruza hacia abajo el límite inferior de la nube, puede ser el inicio de una tendencia bajista, pero esto debe ser confirmado por el RSI, que deberá encontrarse por encima del nivel de sobreventa. No es recomendable operar en momentos de sobreventa, ya que las probabilidades de una reversión de la tendencia son altas y el riesgo de pérdida mayor.

Por tanto, desarrollar un sistema de trading algorítmico rentable requiere de una correcta selección de indicadores y una combinación adecuada de los mismos. Ichimoku es un indicador clave para identificar tendencias, mientras que el RSI actúa como un complemento para confirmar esas tendencias.

La entrada y salida del sistema vendrá determinada por los cruces con la nube junto a la información complementaria que nos proporciona el RSI. Aunque se han descrito en el documento, se vuelven a mencionar, a modo de recordatorio:

* Entrar largo si: el precio cruza el límite superior de la nube y el RSI se encuentra por debajo de la zona de sobrecompra
* Entrar corto si: el precio cruza el límite inferior de la nube y el RSI se encuentra por encima de la zona de sobreventa
* Salimos del corto/largo: las operaciones cortas y largas son complementarias. Cuando entramos en una operación larga, se está produciendo un cambio de tendencia de bajista a alcista, por tanto, saldremos de una corta. Lo mismo pasa con las operaciones cortas, cuando entramos en una operación corta, estamos cambiando de tendencia alcista a bajista, por tanto, saldremos de una operación larga.

Estos dos indicadores juntos se adaptan a la perfección a mercados tendenciales, por tanto, el objeto de análisis como se ha podido ver a lo largo del documento serán este tipo de mercados.

En concreto, hemos encontrado un set rentable en el mercado del oro, con un timeframe de 4 horas. El set en concreto recordemos es el siguiente:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N.º set | Periodo  TenkanSen |  | Periodo  KijunSen | Periodo  SenkouSpanB | Periodo  RSI | Sobrecompra | Sobreventa |
| 1º | 6 |  | 12 | 67 | 16 | 70 | 30 |

Su gráfica de balance del último tercio:

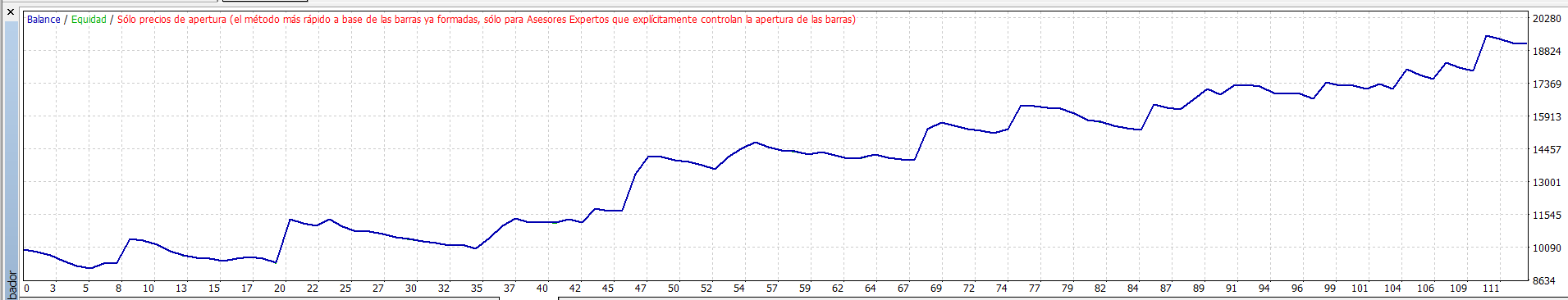


Ilustración 25: Gráfico de balance final del set válido escogido en la última fase de optimización por fuerza bruta, correspondiente al periodo comprendido entre 01/01/2020 – 01/05/2023 para el mercado del oro, con un timeframe de 4 horas. El eje x hace referencia al número de operación, el eje y al balance total

Y sus estadísticas son las siguientes:

* Operaciones: 113
* Beneficio bruto: 19417.58
* Beneficio neto: 9149.70
* Factor de beneficio: 1.89
* Drawdown absoluto: 913.27
* Drawdown máximo: 22.54%
* Drawdown relativo: 22.54%
* Operaciónes ganadoras: 37
* Operaciones perdedoras: 76
* Mayor transacción rentable: 1944.51
* Mayor transacción no rentable: 338.57
* Media de transacción rentable: 524.80
* Media de transacción no rentable: 135.10

El trabajo no termina aquí, pues ahora habría que seguir probando el sistema. Podríamos probar a optimizar por otros parámetros (cambiar la función objetivo), por ejemplo, por drawdown y buscar sets con beneficios, pero que presenten el menor riesgo posible, o buscar un mayor factor de beneficio y comparar los resultados, lo cual, nos dará distintos tipos de estrategias del mismo sistema para operar en el mismo mercado. Además, podemos seguir probando el sistema en distintos mercados para ver si se encuentran más sets no sobreoptimizados y que nos den más posibilidades de operar, ya que podríamos descubrir sets mejores al encontrado. Para ello, se empezaría probando en mercados con comportamiento similar al del oro, donde hemos visto que nuestro sistema funciona bien. Estos mercados podrían ser la plata o el platino, por ejemplo. Primero, empezamos por los timeframes altos y si no encontramos sets buenos, vamos descendiendo. Al terminar con estos mercados, podríamos probar con otros que, en teoría, no se adaptan al sistema, para ver su comportamiento, ya que puede darnos ideas para desarrollar otros sistemas.

Para terminar, como trabajos futuros, contamos con diversas opciones basadas en el sistema desarrollado y el indicador de Ichimoku. En primer lugar, como mencionamos anteriormente, podríamos empezar buscando un set rentable en otro mercado. Una opción podría ser el S&P 500, ya que hemos observado que se adapta bien a nuestro sistema. El proceso para lograrlo sería igual al que se ha descrito en este documento para el mercado del oro.

Posteriormente, podríamos intentar identificar sets rentables en mercados que, a primera vista, no se ajusten a nuestro sistema, como los mercados laterales (EURUSD, por ejemplo).

Podemos mejorar nuestro sistema para que sea más confiable. Por ejemplo, podríamos incorporar una condición en el código que considere la amplitud de la nube (la diferencia entre el límite superior e inferior) y establecer un valor mínimo para dicha amplitud. Esto se debe a que cuanto mayor sea la amplitud de la nube, más sólida será la tendencia en la que nos encontremos. Por tanto, la ruptura de los límites de la nube, tanto al alza como a la baja, nos proporcionará señales más fiables.

Las opciones mencionadas anteriormente surgen a partir del sistema desarrollado, pero también podríamos considerar cambiar el sistema utilizando otros indicadores o modificando sus condiciones de entrada y salida. Por ejemplo, en el caso del indicador de Ichimoku, actualmente solo tenemos en cuenta los cruces con la nube, pero podríamos modificar esto y utilizar otros componentes de dicho indicador. Podríamos incorporar la línea Chinkou de la siguiente manera:

* Cuando Chinkou se encuentre por encima del precio, será una señal de compra.
* Cuando Chinkou se encuentre por debajo del precio, será una señal de venta.

También podríamos determinar posibles puntos de entrada y salida basados en las líneas TenkanSen y KijunSen, de la siguiente manera:

* Cuando TenkanSen cruce al alza a KijunSen, tendremos una señal de compra.
* Cuando KijunSen cruce a la baja a TenkanSen, tendremos una señal de venta.

Como se puede apreciar, el indicador de Ichimoku es muy completo y nos brinda diversas posibilidades para desarrollar sistemas utilizando alternativamente sus componentes.

Las variantes mencionadas anteriormente, son solo una muestra de lo que se podría hacer a partir de los indicadores empleados en el sistema. Existe la oportunidad de explorar y probar nuevos indicadores que se ajusten a diferentes tipos de mercados. Sin embargo, dado el conocimiento adquirido durante el desarrollo de este sistema y las variantes que ofrece, es recomendable iniciar los trabajos posteriores utilizando los conocimientos adquiridos y luego, en una etapa posterior, explorar nuevas posibilidades con otros indicadores y mercados.

**6 - BIB****LIOGRAFÍA**

1. Admirals Markets. Cómo sacar beneficio del indicador de trading RSI. Recuperado de: <https://admiralmarkets.com/es/education/articles/forex-indicators/como-sacar-beneficio-del-indicador-de-trading-rsi> Consultado el 15 de abril de 2023
2. AvaTrade. RSI Trading strategies. Recuperado de: <https://www.avatrade.es/educacion/professional-trading-strategies/rsi-trading-strategies> Consultado el 14 de abril de 2023
3. Carlos Grima Izquierdo. Trading algorítmico desde cero con MetaTrader4 y MQL4
4. Admirals Markets. ¿Qué es Drawdown Forex? Recuperado de: <https://admiralmarkets.com/es/education/articles/forex-basics/que-es-drawdown-forex#:~:text=El%20m%C3%A1ximo%20drawdown%20ser%C3%A1%20la,cada%20uno%20de%20los%20periodos>. Consultado el 17 de abril de 2023
5. EarnForex. Drawdown en el trading de Forex. Recuperado de: <https://www.earnforex.com/es/gu%C3%ADas/drawdown-en-el-trading-de-forex/>

Consultado el 17 de abril de 2023

1. AvaTrade. Drawdown. Recuperado de: <https://www.avatrade.es/educacion/trading-para-principiantes/drawdown> Consultado el 17 de abril de 2023
2. FBS. (s.f.). Métricas de rendimiento en el trading. Recuperado de: <https://fbs.eu/es/analytics/tips/m%C3%A9tricas-de-rendimiento-en-el-trading-29196#:~:text=El%20drawdown%20relativo%20(reducci%C3%B3n%20relativa,relativo%20ser%C3%ADa%20igual%20al%2060%25>. Consultado el 17 de abril de 2023
3. MQL Documentation. iRsi. Recuperado de: <https://docs.mql4.com/indicators/irsi> Consultado el 20 de febrero de 2023
4. MQL Documentation. OrderSend. Recuperado de: <https://docs.mql4.com/trading/ordersend>. Consultado el 22 de febrero de 2023.
5. MQL Documentation. OrderClose. Recuperado de: <https://docs.mql4.com/trading/orderclose>. Consultado el 25 de febrero de 2023
6. MQL Documentation. OrderMagicNumber. Recuperado de: <https://docs.mql4.com/trading/ordermagicnumber> Consultado el 26 de febrero de 2023
7. MQL Documentation. OrderSelect. Recuperado de: <https://docs.mql4.com/trading/orderselect> Consultado el 26 de febrero de 2023
8. Forex.in.rs. Configuración de Ichimoku (7,22,44). Recuperado de: <https://www.forex.in.rs/mercado/configuracion-de-ichimoku-7-22-44/> Consultado el 5 de mayo de 2023
9. MQL Documentation. iIchimoku. Recuperado de: <https://docs.mql4.com/indicators/iichimoku> Consultado el 7 de abril de 2023

**7 – APÉNDICE: CÓDIGO FUENTE DEL SISTEMA**

/\*EA basado en los indicadores de ichimoku y RSI

el sistema entrará y saldrá del mercado

en las siguientes condiciones:

-entrara en largo si: el precio cruza hacia

arriba el límite superior de la nube y

el RSI no está en sobrecompra

-saldrá del largo si: el precio cruza el limite

inferior de la nube hacia abajo y

el RSI no está en sobreventa

-entrara en corto si: el precio cruza el limite

inferior de la nube hacia abajo y

el RSI no está en sobreventa

-saldrá del corto si: el precio cruza el limite

superior de la nube hacia arriba y

el RSI no está en sobrecompra

\*/

#property strict

#define COPYRIGHT "Sergio Bolaños GII"

#define VERSION "1.0"

#define FECHA 24042023

#define NOMBRE "IchimokuRSI"

#property copyright COPYRIGHT

#property version VERSION

//Parámetros Optimizables del EA

//Periodo Tenkan

input uint PeriodoTenkanSen = 9;

//Periodo Kinjun

input uint PeriodoKinJunSen = 26;

//limite Senkou SpanB

input uint PeriodoSenkouSpanB = 52;

//Periodo para el RSI

input uint PeriodoRSI = 14;

//Umbral de Sobrecompra

input uint Sobrecompra = 70;

//Umbral de Sobreventa

input uint Sobreventa = 30;

//Parámetros no Optimizables del EA

//No optimizar

//identificador de operación

input uint Magicnumber = FECHA;

//lotaje por operación

input double lotaje = 0.1;

//Variables globales

//Indica si hay error en los parámetros

bool ErrParam = false;

//identificador operación corta,

//-1 si no hay ninguna

int idcorta;

//identificador operación larga,

//-1 si no hay ninguna

int idlarga;

//tiempo de la última vela procesada

//de esta manera el EA será OpenPrice

datetime TiempoUltimaVelaProcesada;

//Se ejecuta en cada tick

void OnTick(){

//si hay error en los parámetros o

//no nos encontramos en la apertura de la vela

//no hacemos nada

if(ErrParam || Time[0] <= TiempoUltimaVelaProcesada) return;

//si no hay error en los parámetros y

//nos encontramos en la apertura de la vela

//podemos operar

if(!ErrParam && Time[0] > TiempoUltimaVelaProcesada){

//variables para guardar el límite superior

//e inferior de la nube

//en cada caso, esto lo necesitamos

//ya que nuestro sistema tendrá en cuenta

//si el precio está por encima o por debajo

double Lsnube, Linube;

//variables que contendrán el límite superior

//y límite inferior de la nube de ichimoku

double SenkouSpanA, SenkouSpanB;

//variable que contiene el precio de

//cierre de una y dos velas atrás

//respectivamente

double PrecioCierre, PrecioCierre1;

//variable que contiene el valor del RSI

//en la vela anterior

double RSIAnterior;

//Variables para el cálculo del slipagge que vamos

//a permitir en nuestras operaciones

int Slipagge;

double Spread;

//Declaramos las variables que

//van a contener las condiciones

//True si precio > nube

bool condAlcista1 = false;

//True si RSI por debajo de sobrecompra;

bool condAlcista2 = false;

//True si precio < nube;

bool condBajista1 = false;

//True si RSI por encima de sobreventa;

bool condBajista2 = false;

//Si se puede abrir alguna operación

//calculamos las Ichimoku y RSI

if(idlarga == -1 || idcorta == -1){

//Calculamos los valores necesarios

//cálculo del precio de cierre de

//una y dos velas atras

PrecioCierre = Close[1];

PrecioCierre1 = Close[2];

//Calculo de los límites de la nube

//SenkouSpanA y SenkouSpanB

//y del RSI en la vela anterior

SenkouSpanA = iIchimoku(NULL,0,PeriodoTenkanSen,

PeriodoKinJunSen,PeriodoSenkouSpanB,

MODE\_SENKOUSPANA,1);

SenkouSpanB = iIchimoku(NULL,0,PeriodoTenkanSen,

PeriodoKinJunSen,PeriodoSenkouSpanB,

MODE\_SENKOUSPANB,1);

RSIAnterior = iRSI(NULL,0,PeriodoRSI,PRICE\_CLOSE,1);

//Averiguamos las condiciones que se dan

//Antes debemos calcular la nube, la nube

//viene delimitada por el SenkouSpanA y

//el SenkouSpanB, dependiendo cual este por

//encima sabremos cual es el limite

//superior o inferior

if(SenkouSpanA > SenkouSpanB){

Lsnube = SenkouSpanA;

Linube = SenkouSpanB;

}else if(SenkouSpanA < SenkouSpanB){

Lsnube = SenkouSpanB;

Linube = SenkouSpanA;

}else{

Lsnube = SenkouSpanB;

Linube = Lsnube;

}

//Comprobamos si se cumplen

//las condiciones de entrada del sistema

//condiciones alcistas

condAlcista1 = PrecioCierre1 < Lsnube

&& PrecioCierre > Lsnube;

condAlcista2 = RSIAnterior < Sobrecompra;

//condiciones bajistas

condBajista1 = PrecioCierre1 > Linube

&& PrecioCierre < Linube;

condBajista2 = RSIAnterior > Sobreventa;

//Calculamos el Slipagge máximo que vamos

//a permitir en nuestras operaciones

//El slipagge que vamos a permitir como máximo será 2 veces

//el spread del mercado en el que se opere

//Así damos margen a que si alguna operación entra

//con unas condiciones diferentes a las que se mando

//entre sin problema y no de errores

Spread = (Ask - Bid)\*(MathPow(10 , Digits));

Spread = NormalizeDouble(Spread ,Digits);

Slipagge = (int)Spread\*2;

//si se cumplen las dos condiciones alcistas definidas

//y no hay otra operación larga abierta previamente

//abrimos una operación larga

if((condAlcista1 && condAlcista2) && idlarga == -1)

idlarga= AbrirOperacion(OP\_BUY,lotaje,Slipagge,Magicnumber);

//si se cumplen las dos condiciones bajistas definidas

//y no hay una operación corta abierta previamente

//abrimos una operación corta

if((condBajista1 && condBajista2) && idcorta == -1)

idcorta= AbrirOperacion(OP\_SELL,lotaje,Slipagge,Magicnumber);

}

//si existe alguna operación abierta

//entonces cerramos la operación que corresponda

//según las condiciones del mercado en ese momento

if(idcorta != -1 || idlarga != -1){

//si se cumplen las dos condiciones

//alcistas que tenemos definidas

//y hay una operación corta abierta

//cerramos la operación corta

if((condAlcista1 && condAlcista2) && idcorta != -1)

idcorta = CerrarOperacion(Ask , lotaje , idcorta);

//si se cumplen las dos condiciones

//bajistas que tenemos definidas

//y hay una operación larga abierta

//cerramos la operación larga

if((condBajista1 && condBajista2) && idlarga != -1)

idlarga = CerrarOperacion(Bid , lotaje , idlarga);

}//fin del if(No hay operaciones abiertas)

}

}//Fin de la función OnTick

//Abre una operación y comprueba si hay alguna operación fantasma

//Parámetros: Lotaje: El lotaje de la operación

// Tipo: El tipo de operación (compra o venta)

// SM: Máximo slipagge permitido

// idmagic: El magicnumber de esa operación

//Retorno: Devuelve el identificador de la operación

//si la ha podido abrir sin problemas

//o -1 en caso de que exista algún tipo de error.

//Precondición: Para abrir una operación, ya sea compra o venta,

//no debe haber abierta una operación de ese mismo tipo y

//deben cumplirse las condiciones necesarias en el mercado

//para poder abrirlas

int AbrirOperacion(int tipo,double lotes,int SM,int idmagic){

double apertura;//precio de apertura de la operación

int id;//valor de retorno al abrir la operación

//si la operación es una compra (tipo == OP\_BUY)

//el precio de apertura es el Ask

//si la operación es una venta (tipo == OP\_SELL)

//el precio de apertura es el Bid

if(tipo == OP\_BUY){

apertura = Ask;

}else{

apertura = Bid;

}

//abre una operación y guarda el valor en la variable id

//si se abre correctamente se guarda el valor que devuelve OrderSend

//si no se abre correctamente devuelve -1 y

//se vuelve a intentar en el siguiente tick

id = OrderSend(Symbol(),tipo,lotaje,apertura,SM,0,0,NULL,idmagic);

//esperamos por una posible operación fantasma

if(id == -1) GetOperacionFantasma(GetLastError());

return id;

}//Fin AbrirOperacion

//Cierra una operación abierta

//Parámetros: salida: Precio de salida de la operación

// si es una venta será precio Ask si no Bid

// lotaje: Lotaje de la operación

// id: El identificador de la operación

//Retorno: devolverá -1 en caso de que se cierre la operación

// permitiendo así que se puedan abrir nuevas operaciones

// o devolverá el id de la operación en caso de que

// no se pueda cerrar (es decir se mantiene abierta)

//Precondición: Para poder cerrar una operación es necesario que

// exista algún tipo de operación abierta

int CerrarOperacion(double salida , double lotes , int id){

int v = id;//valor de retorno al cerrar la operación

bool cerrar = OrderClose(id , lotes , salida , 0);

//si la operación se ha encontrado y

//se ha podido cerrar v = -1

if(cerrar) v = -1;

return v;

}//Fin de CerrarOperacion

//Controla los tipos de error que pueden ocurrir

//al abrir una operación

//Parámetros:

// error: código de error que se genera cuando

// no se puede meter una operación

//Precondición: No se ha podido abrir una operación

//de forma satisfactoria (id operación = -1)

void GetOperacionFantasma(int error){

//si no hay errores no hacemos nada

if(error == ERR\_NO\_ERROR) return;

//si ocurre un error de los siguientes

//es que no ha ocurrido una operación fantasma

//y por tanto no podemos hacer nada

if (error==ERR\_INVALID\_TRADE\_PARAMETERS ||

error==ERR\_NOT\_ENOUGH\_RIGHTS || error==ERR\_TOO\_FREQUENT\_REQUESTS

|| error==ERR\_ACCOUNT\_DISABLED || error==ERR\_INVALID\_ACCOUNT ||

error==ERR\_INVALID\_PRICE || error==ERR\_INVALID\_STOPS ||

error==ERR\_INVALID\_TRADE\_VOLUME || error==ERR\_MARKET\_CLOSED ||

error==ERR\_TRADE\_DISABLED || error==ERR\_NOT\_ENOUGH\_MONEY ||

error==ERR\_PRICE\_CHANGED || error==ERR\_OFF\_QUOTES ||

error==ERR\_REQUOTE || error==ERR\_LONG\_POSITIONS\_ONLY\_ALLOWED ||

error==ERR\_TOO\_MANY\_REQUESTS || error==ERR\_TRADE\_MODIFY\_DENIED ||

error==ERR\_TRADE\_TOO\_MANY\_ORDERS ||

error==ERR\_TRADE\_HEDGE\_PROHIBITED ||

error==ERR\_TRADE\_PROHIBITED\_BY\_FIFO)

return;

//Si no se ha producido ninguno de los errores anteriores

//esperamos a que la operación aparezca

//en MetaTrader y la podamos recuperar en uno de los siguientes ticks

//esperamos 5 ticks, que solo llegaran si hay conexión

for (int i=0; i<5; i++) {

//Se espera a que llegue un nuevo tick

//esta comprobación se hace cada 2 segundos para no saturar

while (RefreshRates()==false) Sleep(2000);

Sleep(1000); //Esperamos en el este tick

}

}//Fin GetOperacionFantasma

//Comprueba si existe alguna operación abierta con

//el magic number proporcionado

//Parámetros: idmagic: magic number de operación

//Retorno: Si no existe ninguna operación con el magic dado devuelve -1

//si existe alguna operación devuelve el identificador de la operación

int OperacionAbierta(int idmagic){

int total = OrdersTotal(); //Ordenes totales abiertas

int r = -1;//valor de retorno de la función

//se busca entre todas las operaciones abiertas en el sistema

//si alguna de ellas coincide con el magic number dado(idmagic)

//si se encuentra una operación con ese magic el valor de retorno

//de la función(r) será el identificador de la operación

//y paramos de buscar

//si no se encuentra el valor de retorno(r) es -1

//que significa que no hay operaciones abiertas y

//por tanto podremos abrir una

for(int i = 0 ; i < total && r == -1 ; i++){

//true si encuentra esa operación, false si no la encuentra

bool exito = OrderSelect(i , SELECT\_BY\_POS , MODE\_TRADES);

//si encuentra la operación y el maginumber

//de esta operación es igual al dado

if(exito && OrderMagicNumber() == idmagic){

//Si la fecha de cerrado es 0 es que hay operación abierta

if(OrderCloseTime() == 0) r = OrderTicket();

}

}

return r;

}//Fin OperacionAbierta