

Más de **6 000+** robots comerciales e indicadores

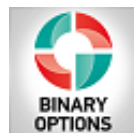
Descargar demo

Comprar

Alquilar

El artículo ha sido enviado al moderador para que lo revise.

Si su artículo ha llamado nuestra atención, muy pronto nuestro moderador se pondrá en contacto con Ud. Podrá discutir con él posibles errores e inexactitudes que pueden estar presentes en su artículo. Siga las actualizaciones en las versiones de su artículo. Cuando el moderador se ponga a estudiarlo, entonces pasará al estado **En revisión**.



Opciones binarias: evaluación de estrategias multidivisa en MT5

editar

[Jose Miguel Soriano](#) | 12 septiembre, 2015

Dado que MT4 no permite, que sepamos, optimizar directamente estrategias basadas en opciones binarias (BOs), desarrollamos en este artículo un sistema sencillo que permite evaluar la bondad de una estrategia en función de sus resultados con opciones binarias. Se escoge MT5 por la mayor potencia y capacidad multidivisa de su probador de estrategias y por la mejor aproximación a la realidad de su algoritmo de generación de ticks. El sistema se aplica sobre cualquier divisa/s o broker. En cada caso, será necesario codificar el núcleo de la estrategia que define cuando comprar o vender.

Introducción

En alguna ocasión hemos recibido encargo de desarrollo de un EA sobre opciones binarias (BOs en adelante). Los brokers que ofrecen este producto instalan sobre MT4 el módulo FXLITE y el usuario puede actuar desde MT4 comprando opciones CALL (al alza) o PUT (a la baja). Si se pulsa F9 no aparecerá la conocida ventana de "abrir orden nueva" sino una más sencilla que permitirá elegir la cantidad a invertir (por ejemplo, 10 € y no 0,10 lotes) y la duración de la opción, normalmente 1, 5, 10, 15, 30 ó 60 minutos.

Independientemente de las particulares, y no públicas, instrucciones que la programación del EA para opciones binarias implica, siempre nos hemos encontrado sin historial para las pruebas y MT4 no las permite en multidivisa. El usuario tiene que colocar el EA en una cuenta demo (normalmente caduca al mes) y "esperar" semanas, conforme ejecuta operaciones en el mercado, para sacar conclusiones. No se puede aprovechar la gran ventaja que sí ofrece una plataforma como MT5: su optimizador...

Como es sabido, una opción binaria es un producto financiero derivado, cuyo subyacente puede ser cualquier otro activo y cuya duración es limitada en el tiempo. Vencido este tiempo, si el precio del subyacente ha subido (opción CALL) o ha bajado (opción PUT), el inversor obtiene un rendimiento de entre el 70% y el 80% (según plazo temporal y broker; normalmente M1 el 80%, H1 el 70%); si el precio ha ido en contra el inversor perderá lo que invirtió. Es decir, si compramos CALL M1 por 10€ y el precio sube tras los 60 segundos, recuperamos 18€ (10€ más 80% de beneficio); si el precio ha bajado, perdemos los 10€.

No hay Take Profit, no hay Stop Loss ni cualquier otro mecanismo de reducción del riesgo. La operación se cierra por vencimiento del plazo temporal. No hay spread; los precios que aplica el broker son los del gráfico: precios BID.

Planteamiento del algoritmo

El sistema de inversión expuesto puede parecer una "lotería"... pero matemáticamente implica que, si queremos un sistema ganador, su esperanza matemática debe ser positiva. Es decir, la estrategia (use ADX, acción del precio, MACD o el Santo Grial) gestiona un sistema donde se pierde todo o se gana el 75% (escogemos esta cifra como media) y en la cual una esperanza matemática mayor que cero exige un porcentaje de operaciones ganadoras tal que...

```
Esperanza matemática= (75 x %Winner) - (100 x %Loss) > 0
%Loss= 100-%Winner ==> 175 x %Winner > 10000 ==> %Winner > 57,14%
```

Es decir, si la estrategia no obtiene al menos un 57,20 % de operaciones ganadoras el sistema será perdedor. Obviamos el uso de sistemas martingala que poco aportan como solución, a nuestro entender.

En un inciso, apuntamos que los indicadores tendenciales suelen producir un porcentaje ganador del 35 al 45%. Los osciladores, por aquello de que el mercado está en rango un 70-80% del tiempo, pueden dar hasta un 60%, y ello, jugando con SL amplios, TP reducidos y otros "complementos". Pero nuestra finalidad, aquí, es la de exponer un mecanismo que nos permita averiguar que porcentaje de operaciones ganadoras dará nuestro sistema sobre BO según el grupo de parámetros de entrada que apliquemos.

Sobre MT5 codificamos la estrategia a usar. Los pasos del algoritmo a desarrollar son:

1. El usuario define el valor de la opción binaria, el plazo temporal y el porcentaje de ganancia que genera la BO.
2. Las posiciones se abren sobre el subyacente de la BO, por ejemplo EURUSD para analizar EURUSDbo.
3. Las posiciones que abrirá el EA serán de una sola transacción y con lote mínimo para obviar el efecto de grandes caídas (draw down) y el efecto del Stop Out.
4. Almacenaremos el precio BID al abrir posición, sea BUY o SELL.
5. La posición se cerrará transcurrido el plazo temporal de BO.
6. El resultado será positivo o negativo por comparación con precio BID al cierre.
7. En OnTrade(), si el resultado es negativo, una variable acumulará el valor de BO con valor negativo.
8. Si el resultado es positivo, la variable acumulará el 75% del valor BO con valor positivo.
9. Igualmente, en OnTrade() iremos acumulando otros datos para información estadística.
10. La función OnTester() entregará el valor final de esa variable o de otros datos estadísticos, para poder optimizar el EA por el resultado de la inversión en BO y no por los resultados en el subyacente.
11. A través de Print() en OnInit() le daremos al usuario más información sobre porcentajes u otros datos que reclame.

Desarrollo de la solución

En nuestro anterior [artículo](#) ya esbozamos nuestro sistema multidivisa. Partimos, pues, de dicho sistema y tendremos...

```
input string cadParesFX= "EURUSD;EURJPY;GBPCHE"; //cadena que introduce el usuario al carga
```

```

string arrParesFX[];    //array que contendrá el nombre de los pares con que trabajará el EA
int nSimbs= 1;          //variable que informa a todo el programa del número de símbolos con los que trabaja

int OnInit()
{
    .../...
    nSimbs= cargaPares(cadParesFX, arrParesFX);    //devuelve array arrParesFX con los pares a trabajar EA
    .../...
}

```

donde nSimbs informará a todo el EA del número de símbolos con los que se trabaja. arrParesFX es una matriz en la que guardamos el nombre de cada símbolo de los definidos en el parámetros de entrada cadParesFX.

A continuación podemos hacer numerosas comprobaciones en relación a la cadena que introdujo el usuario. Por ejemplo, previniendo usuarios inexpertos, ¿están todos los símbolos seleccionados en la ventana de mercado?

```

int OnInit()
{
    .../...
    for(k= 0; k<nSimbs; k++) confirmSimbMarketWatch(nombreParesFX[k]);
    .../...
}

```

```

//----- AÑADE UN SÍMBOLO A LA VENTANA DE MERCADO market watch -----
bool confirmSimbMarketWatch(string simb)
{
    bool    confirma= simb!="";
    int     i= 0, total= 0;
    string  nombSimb= "";
    if(confirma)
    {
        confirma= false;
        total= SymbolsTotal(false);    //Total de simbs en el servidor
        while(!confirma && i<total)
        {
            nombSimb= SymbolName(i, false);
            if(nombSimb==simb) confirma= SymbolSelect(nombSimb, true);    //Lo añade a la ventana de mercado
            i++;
        }
    }
}

```

```

    }
    return (confirma); //informa si algún símbolo no está disponible
}

```

Esta función comprueba si los símbolos solicitados están en la ventana de mercado, y, si no lo están pero el broker sí los sirve, los selecciona y coloca en dicha ventana. Igual podríamos comprobar con los gráficos abiertos.

Como tratamos de evaluar opciones binarias (BOs), al usuario le pediremos como parámetros de entrada el valor de la opción binaria, su duración temporal y el porcentaje de ganancia si la operación resulta ganadora.

```

enum mis_TMP_OpcBin {_M1bo=1, _M5bo=5, _M10bo=10, _M15bo=15, _M30bo=30, _H1bo=60 };

//...
input mis_TMP_OpcBin minutosPosicViva = 1; //Minutos de apertura de la posición
input double valorBO = 5; //valor de la opción binaria

```

```

input double porcGananc = 75; //porcentaje de ganancia en BO
//...

```

Con la enumeración mis_TMP_OpcBin ceñimos al usuario en la elección del plazo temporal. Hemos cubierto, pues, el paso 1 del algoritmo descrito al inicio. El porcentaje de ganancia se podría programar para cada tipo de BO seleccionada, pero depende del broker y, a los efectos, poco cambia lo aquí expuesto.

Los pasos 2 y 3 son rutinarios en cualquier EA y no se comentan; dependerán entre otras cosas, de la estrategia de compra / venta a aplicar.

Para cumplir con el paso 4 (almacenar el precio BID al abrir posición, sea BUY o SELL) y en multidivisa definimos como variable pública global las matrices arrPrecApertPosic[] y arrTmpTransPrev[...]

```

//...
double arrPrecApertPosic[],
      arrTmpTransPrev[];
//...
void OnInit()
{
    //...
    ArrayResize(arrPrecApertPosic, nSimbs);
    ZeroMemory(arrPrecApertPosic);
    ArrayResize(arrTmpTransPrev, nSimbs);
    ZeroMemory(arrTmpTransPrev);
    //...
}

```

```
}

```

De manera que si el usuario definió la cadena "EURUSD;EURJPY;GBPCHF" (porque quiere evaluar "EURUSDbo;EURJPYbo;GBPCHFbo"), cada vez que nuestra estrategia ordene abrir posición en EURUSD, sea BUY o SELL, arrPrecApertPosic[0] guardará el precio BID de EURUSD; arrPrecApertPosic[1] lo hará para EURJPY y arrPrecApertPosic[2] para GBPCHF. Pero esa numeración no es algo que nos deba preocupar; tendremos una función que nos busque en la matriz de nombres de símbolos y nos devuelva el índice del símbolo que buscamos... Igual ocurrirá con las horas de cierre de posición de cada símbolo: se guardarán en arrTmpTransPrev[.] para posterior comparación con la hora de cierre actual y detectar un nuevo cierre en el símbolo dado.

```
//...
string simb= "EURUSD";
int codS= buscaCadArray(simb, arrParesFX);
//...
arrPrecApertPosic[codS]= precioBID(simb);
//...
```

Dejamos como práctica al lector el definir las dos funciones que aparecen arriba.

Hemos cumplido el paso 4 del algoritmo... ¿Cómo cumplimos el paso 5?, "cerrar posición al concluir el plazo temporal".

```
//...
bool cerrarPosic= cierreHorasApert(simb, minutosPosicViva, _MIN);
//...
```

```
bool cierreHorasApert(string simb, double tmpPosicAbierta, mi_FECHA_HORA tmp)
{
    bool resp= false;
    int codS= 0;
    double horasPosic=0, benef= 0;
    if(tmpPosicAbierta>0)
    {
        horasPosic= (double)tiempoNetoPosic(simb, tmp);
        resp= PositionSelect(simb) && horasPosic>horasPosicAbierta;
    }
    return(resp);
}
```

Donde la función tiempoNetoPosic() nos informa del tiempo efectivo de mercado que la posición lleva abierta, obviando las horas de mercado cerrado.

```
double tiempoNetoPosic(string simb, mi_FECHA_HORA modo= _SEG)
{
    datetime arTmpBarra[],
        horaIni= (datetime)PositionGetInteger(POSITION_TIME),
        horaFin= TimeCurrent();
    double contTmp= horaIni>0? CopyTime(simb, PERIOD_M1, horaIni, horaFin, arTmpBarra): -1;
    if(contTmp>0)
    {
        contTmp= contTmp*60; //+(horaFin-arTmpBarra[(int)contTmp-1])+(horaIni-arTmpBarra[0])
        switch(modo)
        {
            case _DIA : contTmp= contTmp/(60*60*24); break; // calcula número de días
            case _HORA : contTmp= contTmp/(60*60); break; // calcula número de horas
            case _MIN : contTmp= contTmp/60; break; // calcula número de minutos
            case _SEG : contTmp= contTmp; break; // calcula segundos
        }
    }
    return(contTmp); //devuelve -1 si ha habido error
}
```

Por tanto, la variable cerrarPosic nos informará cuándo hacerlo por haber transcurrido el tiempo que el usuario definió en "minutosPosicViva" como plazo temporal de la opción binaria.

En todas las funciones se pasa como parámetro la variable "simb"... porque tratamos con un sistema multividua. No nos valen funciones como Symbol().

Entramos en el paso 6 del algoritmo. Exponemos a continuación el código que usaremos en OnTrade(), código que es el núcleo del sistema propuesto.

Como sabemos, OnTrade() es reclamada por el evento que se produce tras cada transacción y, en particular, tras el cierre de una posición, transacción que será siempre del tipo DEAL_ENTRY_OUT, sea la posición BUY o SELL.

```
void OnTrade()
{
    double resultBO= 0, //acumula el resultado de las BOs
        precioCierr= 0; //recogerá precio BID al cierre de la posición
    bool resultPosit= false;
    ENUM_DEAL_TYPE tipoTrans= WRONG_VALUE;
    datetime tmpTransAct= 0; //recoge la hora de la transaccion de cierre actual
    int codS= 0;
    string simb= nSimbs>1? simbNsimasTransHistorial(1, DEAL_ENTRY_OUT): Symbol(); //símbolo de la n-sima
    tmpTransAct= horaNsimasTransHistorial(simb, 1, DEAL_ENTRY_OUT); //hora n-sima transacci
```

```

codS= buscaCadArray(simb, nombreParesFX);
if(arrTmpTransPrev[codS]<tmpTransAct)
{
    tipoTrans= tipoNsimTransHistorial(simb, 1);
    precioCierr= precioBID(simb);
    if(tipoTrans==DEAL_TYPE_BUY) resultPosit= (arrPrecApertPosic[codS]-precioCierr)>0; //se cerró con
    else if(tipoTrans==DEAL_TYPE_SELL) resultPosit= (precioCierr-arrPrecApertPosic[codS])>0; //se entró
    resultBO= resultPosit? valorBOint*porcGananc/100: -valorBOint;
    resultBoEA= resultBoEA + resultBO;
    Print("result BO ", DaS(resultBO, 2));
    if(resultPosit) arrResultBO[1]++;
    else arrResultBO[0]++;
    arrTmpTransPrev[codS]= tmpTransAct;
}
return;
}

```

En la función expuesta, después de cumplir con lo que se nos pedía en los exámenes de la Universidad (definir e inicializar al inicio del procedimiento las variables que intervienen en el mismo), identificamos el símbolo de la última posición cerrada con la función `simbNsimTransHistorial()`...

```

//----- SIMBOLO n TRANSACCIÓN del HISTORIAL empezando por el final -----
string simbNsimTransHistorial(int numTrans= 1, ENUM_DEAL_ENTRY tipoBusca= WRONG_VALUE)
{
    //contando desde el final
    int i, nTotal= 0, cuenta= 0;
    ulong ticket= 0;
    ENUM_DEAL_ENTRY tipo= DEAL_ENTRY_IN;
    string simb= "";
    if(HistorySelect(0, TimeCurrent()))
    {
        nTotal= HistoryDealsTotal();
        for(i= nTotal-1; i>=0; i--)
        {
            ticket= HistoryDealGetTicket(i);
            tipo= tipoBusca==WRONG_VALUE? tipoBusca: (ENUM_DEAL_ENTRY)HistoryDealGetInteger(ticket, DEAL_ENTRY_TYPE);
            simb= HistoryDealGetString(ticket, DEAL_SYMBOL); //si no se especifica tipo, busca correlato
            if(tipo==tipoBusca) cuenta++;
            if(cuenta==numTrans) break;
        }
    }
    return(simb);
}

```

En este tipo de funciones contamos la última transacción del historial como la primera que buscamos; recorremos el historial desde el final al principio.

Conocido el símbolo, con la función `horaNsimTransHistorial()` identificamos la hora de la transacción de cierre. Esta función sigue un mecanismo semejante a la expuesta arriba y dejamos al lector su resolución. Si esta hora es posterior a la guardada en "arrTmpTransPrev[codS]" sabemos que se ha cerrado una nueva posición en el símbolo dado y pasamos a acumular resultados y toda la estadística que queramos extraer de la operación... ¡de la opción binaria!. El resultado de la posición cerrada no nos importa puesto que el spread lo ha distorsionado; nos interesa sólo el precio BID al que cerró.

Tras comparar este precio BID de cierre con el valor que teníamos almacenado en `arrPrecApertPosic[]`, el código...

```
precioCierr= precioBID(simb);
if(tipoPosic==DEAL_TYPE_BUY) resultPosit= (arrPrecApertPosic[codS]-precioCierr)>0;          //PUT
else if(tipoPosic==DEAL_TYPE_SELL) resultPosit= (precioCierr-arrPrecApertPosic[codS])>0;      //CALL
resultBO= resultPosit? valorBO*porcGananc/100: -valorBO;
resultBoEA= resultBoEA + resultBO;
```

Nos resuelve la incógnita que buscamos: "resultBO" ha calculado en cada cierre qué cantidad ganábamos o perdíamos y "resultBoEA" ha ido acumulando los valores de "resultBO" tras cada cierre. Por supuesto, "resultBoEA" y "arresultBO[]" habrán sido definidas como variables globales públicas al inicio del código.

En dicho código, "tipoTrans" ha sido identificado con una función de desarrollo semejante a la expuesta dos cuadros más arriba: `tipoNsimTransHistorial()`.

Con la expresión...

```
tmpTransPrev[codS]= tmpTransAct;
```

guardamos la hora actual de cierre para compararla con la del siguiente cierre que se produzca en el símbolo.

Hemos satisfecho de esta manera el paso 9.

El paso 10 lo resolvemos en OnTester()...

```
double OnTester()
{
    double resp= 0;
    resp= resultBoEA;
    return(resp);
}
```


De manera que al optimizar podremos pedirle al optimizador que evalúe por el resultado de la estrategia en las opciones binarias y no en el subyacente. La variable "resp" es la que servirá como criterio de optimización si en la ventana de configuración del optimizador elegimos "Custom max". En OnTrade() podemos recoger todos los datos estadísticos que imaginemos, por ejemplo, si pensamos aplicar martingala, la media y el máximo de resultados negativos consecutivos nos puede venir bien. Y si aplicamos el siguiente código...

```
enum mis_RATIOS_OPT {RATIO1, RATIO2, RATIO3};
//...
input mis_RATIOS_OPT ratioOptim= RATIO1;
//...

double OnTester()
{
    double resp= 0;
    switch(ratioOptim)
    {
        case RATIO1: resp= resultBoEA; break;
        case RATIO2: resp= ...; break;
        case RATIO3: resp= ...;
    }
    return(resp);
}
```

El usuario podrá seleccionar el ratio por el que quiere optimizar el EA.

Completados todos los pasos del algoritmo, podemos darle al usuario una primera información en OnDeinit()...

```
void OnDeinit(const int motivo)
{
    //...
    Print("EA resultBo ", DoubleToString(resultBoEA, 2));
    Print("total nº BO ", DoubleToString(arrResultBO[1]+arrResultBO[0], 0));
    Print("%Winner ", DoubleToString(arrResultBO[1]/(arrResultBO[1]+arrResultBO[0])*100, 2));
    Print("%Losser ", DoubleToString(arrResultBO[0]/(arrResultBO[1]+arrResultBO[0])*100, 2));
}
//...
return;
}
```

Con esa simple información, el usuario sabe ya si su estrategia es tan maravillosa como le pareció apreciar en los dos primeros días de ejecución en la cuenta demo de su broker de BOs y... volviendo al inicio del artículo, si "% Ganador" no es superior al 57,2% (al 60% redondeando) siempre perderá dinero, por muchas maravillas que haga su EA, teniendo en cuenta que los resultados de

optimización difícilmente se consiguen en ejecución real.

Conclusión

No conocemos la oferta de todos los brokers que ofrecen BO, pero creemos que el sistema expuesto resuelve los problemas que MT4 nos ha planteado en este tipo de activos...

Hemos visto como obtener el resultado del EA por los criterios de la BO y no por los del activo subyacente. MT5 nos informa de los precios BID de apertura y cierre de las posiciones y la comparación entre estos precios nos dice si BO ha sido rentable o nos ha hecho perder. Todo este mecanismo lo implementamos mediante matrices dinámicas en un sistema multidivisa que nos permitirá, por ejemplo, analizar consecuencias de invertir en BO sobre subyacentes correlacionados positiva o negativamente. Es decir, con la cadena "EURUSD;USDJPY;GBPUSD" podremos sacar conclusiones sobre la inversión en "EURUSDbo;USDJPYbo;GBPUSDbo". El sistema funcionará igual en monodivisa; la cadena "EURUSD" definirá todas las matrices como dimensión uno.

El potente sistema de optimización de MT5 nos permitirá sacar conclusiones que difícilmente podría proporcionar MT4, si bien es verdad que casi todo el sistema expuesto podríamos haberlo codificado en mql4, aunque, siendo puristas, mql4 no nos permite un mecanismo exacto de multidivisa al tener que buscar la información de evento de tick desde OnTimer() hacia OnChartEvent().

Suerte con las "peligrosas" opciones binarias y los errores en los que se pueda haber incurrido a lo largo de esta exposición ruego me sean informados... y perdonados.

Únete a nosotros, idescarga MetaTrader 5!

[Windows](#)

[iPhone/iPad](#)

[Mac OS](#)

[Android](#)

[Linux](#)

[Lenguaje de estrategias MQL5](#) | [Biblioteca del código fuente](#) | [Cómo escribir un Asesor Experto o un Indicador](#) | [Encargar el desarrollo del Asesor Experto](#)

[Descargar MetaTrader 5](#) | [Plataforma comercial MetaTrader 5](#) | [Tienda de aplicaciones](#) | [MQL5 Cloud Network](#)

[Acerca del proyecto](#) | [Historial del sitio web](#) | [Condiciones del uso](#) | [Política de confidencialidad](#) | [Contactos](#)

Copyright 2000-2015, MQL5 Ltd.