GECCO 2011 Industrial Challenge: Optimizing Foreign Exchange Trading Strategies
Oliver Flasch, Thomas Bartz-Beielstein 1, Wolfgang Kantschik, and Christian von Strachwitz 2

March 15, 2011

The goal of the GECCO 2011 Industrial Challenge is to find profitable trading strategies for automatic foreign exchange trading. A trading strategy is a function that generates a trading signal based on exchange rate time series data. Trading strategies are assessed by their cost-corrected profitability and by their risk. This document provides a set of rules and regulations, a detailed problem description, as well as contact and submission information.

1 Introduction

The foreign exchange (FX) market is a decentralized over-the-counter financial market for trading currencies to enable international trade and investment. As the largest and most liquid financial market in the world, it is open for trading 24 hours on every weekday and globally available [Weithers, 2006]. Its main participants are not only financial institutions such as banks, central banks, and hedge funds, but also private brokers and large industrial corporations. High frequency intraday FX trading is very common and completely automated through algorithmic trading systems. Because profitable intraday trading is a very challenging problem, this competition focuses on algorithmic FX trading based on hourly data.

Currencies can be traded via a wide variety of different financial instruments, ranging from simple spot trades over to highly complex derivatives. In this competition, a simplified model of the FX market is used to define a simple but still realistic test problem, which will be referred to as the FX problem in the remainder of this document. Participants have to solve this problem by generating trading signals for a representative set of currency pairs. These trading signals are assessed by a quality measure that takes profitability (incorporating trading costs) and risk into account.

The FX problem belongs to the class of time series forecasting problems [Brockwell and Davis, 2002]. Although many different methods can be used for time series forecasting, Computational Intelligence (CI) methods, such as Evolutionary Computation, are an attractive option. CI methods are robust to changes in the underlying system, i.e. changes in the market conditions, which can lead to reduced risk and lower maintenance costs. As an example, Genetic Programming (GP) has been successfully applied to the FX problem, both in research and in industry [Dempster and Romahi, 2002, Wilson and Banzhaf, 2010, 2009, Bhattacharyya et al., 2002]
Austin et al., 2004]. These findings make CI-based systems an interesting alternative to the classical time series analysis methods more widely applied in quantitative finance, and motivated this competition [Hamilton, 1994]. Highlights of the GECCO 2011 Industrial Challenge include:

- **Interesting Problem Domain:** The FX market, with its complex patterns and behavior, offers a fascinating test case for innovative optimization methods.
- **Real-world Data:** Multiple real intraday FX return data sets are provided for training and testing trading strategies.
- **Realistic Quality Measurement:** Trading strategies are scored using a simple but realistic trading simulator that takes trading costs and risk into account.
- **Fair Submission Assessment:** Winners are determined in a second round by simulated trading based on new FX return data sets, rendering accidental overfitting or cheating impossible.

The remainder of this document gives the information needed to take part in this competition and is organized in three parts: Section 2 introduces the problem of finding strategies for algorithmic trading in the FX market, its related objective function, and the provided training and test data sets. Section 3 presents the set of rules and regulations. Finally, Section 4 gives information on how to participate in the challenge.

## 2 Problem Description

The objective of this competition is to find profitable trading strategies for the FX market. A trading strategy \( f \) is a function generating a trading signal \( S \) based on a FX return time series \( R \):

\[
\begin{align*}
  f(R) \rightarrow S
\end{align*}
\]  

A FX return time series \( R \) is a time series of absolute returns, i.e. absolute exchange rate changes, for a currency pair. The first currency in a currency pair is known as the base currency, the second currency is called the counter currency. A single data point \( R(t) \) in a FX return time series \( R \) shows the absolute value change of the counter currency in relation to the base currency, that occurred in the time interval represented by the data point at \( t \). For example, the data point “2010-03-31 10:00:00 0.0016” of the hourly FX return time series “EURUSD” encodes that, in the time interval from 10:00:00 to 10:59:59, the base currency (EUR) has increased in relative value by 0.0016 USD.

A trading signal is a time series of Long (represented as the integer 1), Short (-1), or Flat (0) signals that defines, for each point in time,
whether to allocate a fixed size amount of money to the base currency (Long position in the base currency), to allocate that amount to the counter currency (Short position in the base currency), or to stay out of market (Flat signal). Short positions generate profits when the base currency loses in relative value, while Long positions generate profits when the base currency gains in relative value. The quality of a trading signal is assessed by simulated trading and analysis of the generated returns, as described in the next subsection and shown in Listing 1.

2.1 Objective Function

Assessing the quality of a trading signal $S$ for a given training or test FX return time series $R$ is a two-step process:

1. The trading signal return time series $T_{(R,S)}$, i.e. the cost-corrected profits and losses, generated by trading the currency pair of $R$ based on the trading signal $S$, is calculated.

2. The quality of $S$ is then given by the simplified Sharpe Ratio\(^3\) of $T_{(R,S)}$.

\[ \text{SignalQuality}(R,S,t_c) := \text{SharpeRatio}[	ext{SimulateTrading}(R,S,t_c)] \]

\[ = \frac{\text{mean}[	ext{SimulateTrading}(R,S,t_c)]}{\text{SD}[	ext{SimulateTrading}(R,S,t_c)]} \]
SimulateTrading ← function(R, S, t_c) {
    l ← length(R)
    T_{(R,S)} ← numeric(l) # initialize T_{(R,S)} with l zeros
    P ← S[1] # current position

    for (i in 2:l) { # for i from 2 to l...
        # if in FLAT position...
        if (P == 0 && S[i] == 0) {
            # in FLAT position, next signal is FLAT...
            # (no state change)
        } else if (P == 0 && S[i] == 1) {
            # in FLAT position, next signal is LONG...
            P ← 1 # Go LONG.
        } else if (P == 0 && S[i] == -1) {
            # in FLAT position, next signal is SHORT...
            P ← -1 # Go SHORT.
        } # if in LONG position...
        else if (P == 1 && S[i] == 1) {
            # in LONG position, next signal is LONG...
            T_{(R,S)}[i] ← R[i] # Record return.
        } else if (P == 1 && S[i] == 0) {
            # in LONG position, next signal is FLAT...
            T_{(R,S)}[i] ← R[i] - t_c # Record return minus costs.
            P ← 0 # Exit position.
        } else if (P == 1 && S[i] == -1) {
            # in LONG position, next signal is SHORT...
            T_{(R,S)}[i] ← R[i] - t_c # Record return minus costs.
            P ← -1 # Go SHORT.
        } # if in SHORT position...
        else if (P == -1 && S[i] == -1) {
            # in SHORT position, next signal is SHORT...
            T_{(R,S)}[i] ← -R[i] # Record return.
        } else if (P == -1 && S[i] == 0) {
            # in SHORT position, next signal is FLAT...
            T_{(R,S)}[i] ← -R[i] - t_c # Record return minus costs.
            P ← 0 # Exit position.
        } else if (P == -1 && S[i] == 1) {
            # in SHORT position, next signal is LONG...
            T_{(R,S)}[i] ← -R[i] - t_c # Record return minus costs.
            P ← 1 # Go LONG.
        }
    }
    return(T_{(R,S)})
}
The score of a submission is defined as the sum of the signal qualities over the three test data sets given in Section 2.2. A high-level overview of the trading strategy generation and assessment process is given in Figure 2. Implementations of the complete objective function including both steps are available in R and Java. See Section 4 for instructions on how to download reference material, source code, and documentation.

### 2.2 Training- and Test-Datasets

Training- and test time series data for the FX problem consist of 3900 data records from three different currency pairs. For the first currency pair (AUDUSD), data of the year 2005 is provided, while 2010 data is provided for the remaining two currency pairs (EURUSD and GBPUSD). As the score of a trading strategy is obtained by taking the sum of the signal qualities over the three test data sets, signal strategies that are robust to a wide variety of market conditions have an advantage. The data sets with their respective time intervals and sizes are shown in Table 1.

<table>
<thead>
<tr>
<th>Currency Pair</th>
<th>Start Date</th>
<th>End Date</th>
<th># Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDUSD Training</td>
<td>2005-04-01 00:00</td>
<td>2005-06-09 23:00</td>
<td>1188</td>
</tr>
<tr>
<td>AUDUSD Test</td>
<td>2005-06-10 00:00</td>
<td>2005-06-30 23:00</td>
<td>357</td>
</tr>
<tr>
<td>EURUSD Training</td>
<td>2010-04-01 00:00</td>
<td>2010-06-10 23:00</td>
<td>1185</td>
</tr>
<tr>
<td>EURUSD Test</td>
<td>2010-06-11 00:00</td>
<td>2010-06-30 23:00</td>
<td>324</td>
</tr>
<tr>
<td>GBPUSD Training</td>
<td>2010-04-01 00:00</td>
<td>2010-06-10 23:00</td>
<td>1185</td>
</tr>
<tr>
<td>GBPUSD Test</td>
<td>2010-06-11 00:00</td>
<td>2010-06-30 23:00</td>
<td>324</td>
</tr>
</tbody>
</table>

Trading costs as defined in Section 2 are given in Table 2. These costs, given in counter currency units, apply each time a trade is

---

4 The EURUSD and GBPUSD FX return time series are based on the same time intervals to enable trading strategies that exploit possible lagged correlations of these data sets.
exit (see Section 2). Figure 3 shows a plot of the accumulated FX return time series AUDUSD, to give an idea of the data. Note the individual returns have been accumulated in this plot. All data sets are made available in CSV format. Section 4 gives instructions on how to obtain these data sets.

<table>
<thead>
<tr>
<th>Currency</th>
<th>Trading Cost (USD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDUSD</td>
<td>0.0002</td>
</tr>
<tr>
<td>EURUSD</td>
<td>0.00015</td>
</tr>
<tr>
<td>GBPUSD</td>
<td>0.0003</td>
</tr>
</tbody>
</table>

Table 2: Trading cost (given in counter currency units) for each currency pair.

![AUDUSD](image)

Figure 3: The accumulated FX return data set AUDUSD. Shown are both training and test data ranges.

3 Rules and Regulations

The challenge is organized in two rounds. In the first round, participants submit their score as defined in Section 2.1. These scores are based on the three test data sets given in Section 2.2. They also submit their executable trading strategy generator together with a two page report describing their approach. Submissions are ranked by score. In the second and final round, the ten best submissions of the first round will be ranked by their scores computed on test data sets collected after the submission date, for the same three currency pairs as used in the first round. To be evaluated in the final round of the competition, the submitted software must be able to accept training and test data in CSV format. It must be able to return trading signals for the test data sets in the form of CSV files.\(^5\) The overall runtime for training per currency pair must not exceed 4 hours on a modern compute server.\(^6\) The software must be executable in Ubuntu Linux Server 10.4 LTS or Microsoft Windows.

\(^5\) The concrete CSV format of data and signal files must match the examples provided by the organizers. See Section 4.1 for details.

\(^6\) The compute servers used in the final round are equipped with Intel Xeon E5540 CPUs (2.53 GHz) and 4 GB RAM.
XP SP3. Software written for VM-based platforms that are available for both operating systems will run on a Linux system. The organizers will collaborate with the ten finalists to evaluate their software in the final round. The winner of the GECCO 2011 Industrial Challenge will be the participant with the highest score in the final round.

4 Submission

Submissions to the first round should consist of:

- The score as defined in Section 2.1 based on the test data sets given in Section 2.2,
- executable software for generating trading signals, as described in Section 3,
- short instructions how to run the software,
- a short report on the methods used (maximally two pages).

Please use the form at http://gociop.de/gecco-2011-industrial-challenge/ to send your submission. You can contact the organizers via email (gecco@f10.fh-koeln.de) if you have any questions.

4.1 Software and Data

Implementations of the objective function, accompanying tools, example trading signals, and the training and test data sets for the first round of the competition are available for download at http://gociop.de/gecco-2011-industrial-challenge/.

4.2 Important Dates

- Software and Data Availability: immediately
- GECCO 2011 Industrial Challenge Submission Deadline: June 1, 2011
- Start of Second Round: June 13, 2011

4.3 Organizing Committee

- Thomas Bartz-Beielstein, Cologne University of Applied Sciences
- Oliver Flasch, Cologne University of Applied Sciences
- Wolfgang Kantschik, DIP Dortmund Intelligence Project GmbH
- Christian von Strachwitz, Quaesta Capital GmbH
- Wolfgang Konen, Cologne University of Applied Sciences
- Pier Luca Lanzi, Politecnico di Milano
- Jorn Mehn, Cranfield University
References


