The FX Blender - Multivariate Generation of High-Quality, Real-Time Foreign Exchange Data

ULRICH A. MÜLLER

UAM.2001-06-19

January 18, 2002

An internal paper by the O&A Research Group

Contents

1 Introduction

Foreign exchange (FX) rates are available in real time from different sources. The quality of such data is mixed. For some FX rates, some data sources provide good data with few outliers, small time delays and few data gaps. Some other FX data streams are worse – sometimes very bad.

All data applications benefit from high data quality, but market making is particularly sensitive to bad data. OANDA's FXTrader needs data of very high quality to prevent clients from exploiting pricing errors.

In this document, an algorithm to improve the data quality is described. The algorithm has been implemented by Kris Meissner in a software tool caller FXBlender.

The following ideas help to achieve the goal of high data quality:

- Using different data sources.
- Using mathematical relations between different FX rates. In this document, this means triangular currency arbitrage relations.
- Using a model of data errors based on known facts on the FX markets and data sources.

The FX rates of the FXTrader are defined in a rather small set of major currencies with many triangular currency relations. Triangular arbitrage sets powerful constraints on the rates. A correlation analysis may also be helpful, but it cannot add a lot of value in our case of major FX rates. In other cases such as minor rates and interest rate data, a correlation analysis would be very beneficial.

For each FX rate stream from each source, a univariate outlier filter first cleans the data and removes the outliers, Nevertheless, there is still a rather wide bandwidth of deviating opinions about the current quote value. There are the direct quotes of the three sources (which may not exist in the case of "exotic" cross rates) and some indirect quotes as computed through a third currency, the so-called vehicle currency. All these many opinions are brought into line through weighted averaging. The weight of an individual quote is high if the data source is good and fast, the rate is between major currencies, the quote is recent (rather than stale), the bid-ask spread is low and, for computed cross rates, the base rates are synchronous. The bid-ask spread of the resulting quote also depends on the scattering of individual quotes on the price axis. If these quotes strongly disagree, the resulting spread widens.

The algorithm has beneficial side effects such as the filling of data gaps. Cross rates are produced in acceptable quality even if no source has direct quotes for that currency pair. The produced time series of prices have a unique appearance. They are no mere copies of data from any available source, but they are at least as good as the data from the best source. They are perceived as O & A data with unique characteristics, though similar to the best source.

The result of the analysis is a set of FX data streams that can be used in real time or in historical applications. The set is complete insofar as all possible cross rates between all analyzed currencies can be supported.

This document covers all the algorithmic ideas needed to write a software tool. In fact, a prototype of the tool has been written before the writing of the first draft of this document, and the production tool named FXBlender has been written afterwards. The document does not treat details of the software implementation.

Index j	1	2	3	4	5	6	7
Currency	JPY	CHF	CAD	USD	AUD	GBP	EUR
Error multiplier E_j	1.3	1.6	1.3	1	1.6	1.3	1.15

Table 1: A set of $m = 7$ currencies, in the sequence of increasing likelihood of being the exchanged currency in usual FX rate definitions. Error multiplier values E_j are also proposed.

2 Basic properties of the data

2.1 Basic properties of the data sources

We consider n different data sources. Some of these consist of indicative market maker quotes, some others provide binding quotes for a trading platform. This difference can be seen with regard to two different criteria, the expected data error and the expected delay of quotes.

The expectation of the data error depends on several variables to be modeled step by step. First we define the basic data error due to the nature of the data source. This error exists even if all other variables are most favorable for data quality. The first reason for the basic data error is the granularity of quotes. Each quote has an error band of half a basis point by virtue of the basis point granularity. In case of indicative quotes, the basic data error is increased by a market maker bias (see literature), the desire of individual market makers to attract more buyers than sellers or the other way around. Furthermore, some contributors may simply produce unreliable quotes as these quotes are not binding.

In a first approach, we choose a basic error of $b = 0.00003$ for excellent data sources (UBS-TP), $b = 0.00005$ for the slightly worse source UBS-I and $b = 0.00015$ for the indicative quotes of Reuters. These are relative errors of prices in the best of circumstances. In case of minor currencies, large bid-ask spreads and staleness of prices, the expectation of the relative error can be far greater as to be shown later. As a convention, we sort the sources according to their qualities, so b_0 is the lowest error of the best source.

The delay d of a source is the average time interval between a market situation that triggered a contributor to produce a quote and the actual arrival of the so produced quote in O $ A's data repository. In case of indicative quotes, some contributors post copied quotes to advertise their presence on the market. These copied quotes imply an additional delay. Therefore, the average delay of indicative data sources is estimated to about $d = 15$ seconds (Reuters), whereas the best data source has $d = 0$ (UBS-TP); for UBS-I, we assume a delay of $d = 1$ second.

For some minor rates such as CHF/JPY, Reuters (or one of their contributors) may repeatedly quote stale price values (e.g., CHF/JPY around 4 May 2001). In this case, the true delay d may extend to hours or even days instead of 15 seconds. In order to cover these cases, the minor rates from Reuters should be studied, and the their delays d should be configured to be very large, if necessary.

2.2 Basic properties of the currencies

We consider m different currencies for which some FX rates are available from the data sources. For each data source, the set of currencies must be contiguous. This means that each currency can be exchanged against each other currency by using a direct FX rate or FX rates between the two currencies and a third vehicle currency. In most practical cases, this condition is fulfilled if the US Dollar (USD) is in the set of considered currencies. In Table 1, an example of a currency set is given.

The two currencies of an FX rate have an influence on the data quality and the expected data error. Major FX rates are between major currencies and have a high liquidity and large transaction volumes. In such markets, the competitive pressure of publishing correct prices is high, and pricing biases due to low liquidity are small. If both exchanged currencies are minor, their direct FX market hardly exists, and the data error can be expected to be particularly high. This effect is modeled by multipliers to the base error. Each currency has an error multiplier E. The more "minor" a currency, the larger the multiplier. The multipliers of both currencies of an exchange rate apply. The basic data error of the direct exchange rate between currency j and currency k is

$$b_{i,j,k} \quad = \quad E_j \, E_k \, b_i \tag{2.1}$$

with the two multipliers E_j and E_k and the basic error b_i of the data source i as introduced in Section 2.1. Table 1 has some values of error multipliers. These values originate from practical experience and intuition rather than a scientific study.

The results of Equation 2.1 are satisfactory for reasonable data sources and major rates. As a matter of fact, the data quality of some minor rates as provided by Reuters is very bad, so Equation 2.1 gives to small error estimates. These cases should be studied. The true basic error $b_{i,j,k}$ can be higher by *orders of magnitude* in some extreme cases as CHF/JPY around 4 May 2001. Fortunately, the chosen method of averaging is robust, so the bad quotes from bad sources will not spoil the results.

2.3 Basic properties of the FX rates

FX rates such as EUR/JPY have two currencies:

1. the *exchanged* currency, here EUR;
2. the currency in which the value of one unit of the exchanged currency is *expressed*, also called the *numeraire* currency, here JPY.

FX rate names are constructed using the scheme $<$ exchanged $>$ / $<$ numeraire $>$. In practice, the roles of the currencies in usual FX rate definitions are fixed. Some currencies tend to be used as exchanged currencies whereas others are numeraire currencies in most FX rates. In Table 1, the currencies are sorted according to this criterion. For any currency pair taken from Table 1, the currency lower on the list is the usual exchanged currency.

In the pricing engine, both possible forms of an FX rate, e.g. EUR/JPY and JPY/EUR, are tested. For one of the two forms, the O & A data repository usually has no data. In the very rare case that both forms exist, the pricing engine chooses the form with the higher number of stored ticks. For some minor cross rates, both forms are missing, so the engine will generate the resulting prices only as computed cross rates. Such minor cross rates are called inactive and characterized by an activity indicator $\delta_{i,j,k} = 0$, whereas normal, active FX rates have $\delta_{i,j,k} = 1$. We generally define $\delta_{i,k,j} \equiv \delta_{i,j,k}$ and $\delta_{i,j,j} \equiv 0$ as FX rates between identical currencies do not exist.

The pricing engine uses FX rates in the form $<$ currency$_j$ $>$ / $<$ currency$_k$ $>$ with $j > k$. If the FX rate definitions as found in the O & A data repository do not conform, they have to be inverted, and a boolean variable is used to store the information on whether they are inverted or not. If the scheme of Table 1 is used, inverted currencies will not be found, though.

The following variables of an FX tick are used:

1. the time stamp $t_{i,j,k,\mathrm{now}}$;
2. the bid price $p_{\mathrm{bid},i,j,k,\mathrm{now}}$;
3. the ask price $p_{\mathrm{ask},i,j,k,\mathrm{now}}$;

As in Equation 2.1, the index i means the source, the index j the exchanged currency and k the numeraire currency. The time series index is not explicitly noted, but the subscript "now" indicates the last valid tick of the time series before or at a time t_{now}. The multivariate timing of the time series will be discussed in Section 3.1. Only those ticks are valid that pass a univariate data cleaning filter. Outlier ticks from the raw data do not enter the multivariate algorithm.

The price values are taken in logarithmic form. The logarithmic middle price is

$$x_{i,j,k} = \frac{\log p_{\mathrm{bid},i,j,k} + \log p_{\mathrm{ask},i,j,k}}{2} \tag{2.2}$$

and the relative bid-ask spread is

$$s_{i,j,k} = \log p_{\mathrm{ask},i,j,k} - \log p_{\mathrm{bid},i,j,k} \tag{2.3}$$

These logarithmic definitions are useful especially if the multivariate algorithm is based on price averaging. Since we use robust averaging with the help of a median, prices may also be used in plain form, but the logarithmic form is still helpful for computing cross rates and final bid-ask spreads.

3 The multivariate algorithm

3.1 Synchronization

The pricing engine is developed for real-time applications, but historical time series can also be processed in order to produce superior historical FX data.

Synchronization of time series is no problem in case of a real-time application. Real time is the synchronization principle. FX ticks from all time series are processed in the sequence of their arrival in O & A's data collector, in real time. The time t_{now} is defined as the time of the newest tick.

In historical applications, synchronization has to be organized. The individual time series have irregularly spaced data with different quoting frequencies and occasional data gaps. The synchronization algorithm is simple. For all time series, two ticks are kept in memory. Initially, these are the two most recent valid ticks before or at the start time of the analysis. The first one is called the "current" tick, the second one the "next" tick. Each step of the algorithm starts with a search for the oldest "next" tick among all time series. This tick becomes the new "current" tick of the series, and its time stamp becomes the new t_{now} of the multivariate analysis. A new tick of the same series is retrieved from the repository and takes the role of the "next" tick. Then the multivariate analysis, as to be described below, can be made based on the "current" ticks of all time series. The whole algorithmic step is iteratively repeated afterwards. O & A's ORLA software also provides a framework for data synchronization.

3.2 Computed cross rates

Computed cross rates play an important role in the multivariate analysis, even for those rates where direct market quotes are given. Each FX rate $<\mathrm{currency}_j> / <\mathrm{currency}_k>$, with $j > k$, can be computed through a *vehicle currency* with index h. Thanks to the use of logarithmic prices, the computation is very simple:

$$x_{i,j,k,\mathrm{vehicle}\,h} \;=\; x_{i,j,h} + x_{i,h,k} \tag{3.4}$$

where $h \neq j$ and $h \neq k$. This formula allows for the computation of all cross rates in a basket of currencies, so a full matrix of x values can be supported. The diagonal elements, $x_{i,j,j}$, are trivial and will not be used. The validity of the matrix elements can be vouchsafed by implementing the antisymmetric definition $x_{i,k,j} \equiv -x_{i,j,k}$. Whenever a new value of $x_{i,j,k}$ is computed, the algorithm should always update the symmetric matrix element $x_{i,k,j}$ with the value $-x_{i,j,k}$. The user may omit some currency pairs of the matrix if there is no need for them.

Cross rate calculations can only be done if both rates with the vehicle currency are available from the data source. In other words, the condition is $\delta_{i,j,h}\delta_{i,k,h} = 1$. The index i still defines the data source. We do not compute cross rates across different data sources in our algorithm. Computed cross rates still belong to one of the data sources. (The suitability of this algorithmic guideline may be reconsidered in a future evaluation).

Both FX rates used in the cross rate computation have a time stamp. If the two time stamps differ, an error is made which will be discussed later. Another error may arise from different value dates of the underlying FX rates. Imminent business holidays affect the value date definition. Some rare local holidays may differently affect the value dates of the underlying FX rates, so a cross rate computed according to Equation 3.4 may have an error related to the interest rate differential over the time interval between the two value dates (Jonathan Buchanan, oral communication).

From the point of view of transaction costs, the relative bid-ask spread of the computed cross rate is the sum of the two underlying relative bid-ask spreads: $s_{i,j,k,\mathrm{vehicle}\,h} \;=\; s_{i,j,h} + s_{i,h,k}$, where the s matrix is symmetric: $s_{i,j,k} \equiv s_{i,k,j}$. Whenever a new value of $s_{i,j,k}$ is computed, the symmetric matrix element $s_{i,k,j}$ should immediately be updated with the same value. The addition of spreads corresponds to the transaction costs of an instantaneous round-trip transaction through the vehicle currency. However, we shall use $s_{i,j,k,\mathrm{vehicle}\,h}$ as a measure for the *uncertainty* of the middle price, where it seems unfair to assume a plain addition of the two spreads. When considering the deviation from the true middle price due to both spreads, it is more appropriate to assume that the deviations of the two original rates are uncorrelated. Then the correct formula is

$$s^2_{i,j,k,\mathrm{vehicle}\,h} \;=\; s^2_{i,j,h} + s^2_{i,h,k} \;\; ; \;\; s_{i,j,k,\mathrm{vehicle}\,h} \;=\; \sqrt{s^2_{i,j,h} + s^2_{i,h,k}} \tag{3.5}$$

This formula should be used. It yields a slightly lower spread than the simple addition of spreads.

3.3 The error of a logarithmic middle price

The true market price can be defined as the average transaction price in a very short time window, worldwide. This is not available, but the ticks from all our data sources provide approximate values. The deviation of $x_{i,j,k,\mathrm{now}}$ from the true logarithmic middle market price is called the *error* $\delta x_{i,j,k,\mathrm{now}}$. We do not know this error, but we can estimate its absolute or squared value. The better the data, the smaller the error.

In fact, modeling the error of FX quotes is a central element of the algorithm. The error consists of four ingredients:

1. the basic data error $b_{i,j,k}$ of a time series as defined in Equation 2.1;
2. the error due to the age or even staleness of the quote;

3. the error due to the bid-ask spread uncertainty;
4. the error due to different value dates of FX rates used for cross rate computation, as explained in Section 3.2.

All ticks have a certain age which is the time interval between the time stamp $t_{i,j,k,\text{now}}$ and the analysis time t_{now}. For slow data sources as discussed in Section 2.1, the (average) delay d_i of the source increases the age of the quote. The age of quotes is expressed in minutes here (an arbitrary decision that can be changed, if this is done in a consistent way). For direct FX quotes, the age is

$$A_{i,j,k} \;=\; d_i + t_{\text{now}} - t_{i,j,k,\text{now}} \tag{3.6}$$

For computed cross rates, we define an effective age of

$$\begin{aligned} A_{i,j,k,\text{vehicle}\,h} &= d_i + t_{\text{now}} - \min[t_{i,k,h,\text{now}},\; t_{i,j,h,\text{now}}] \qquad (3.7)\\ &= d_i + t_{\text{now}} - \frac{t_{i,j,h,\text{now}} + t_{i,k,h,\text{now}}}{2} + \frac{|t_{i,k,h,\text{now}} - t_{i,j,h,\text{now}}|}{2} \end{aligned}$$

Thus we take the higher of the two quote ages. The second formula is mathematically equivalent but represents another view. We take the mean age of the two ticks and add an age penalty of the size $0.5\,|t_{i,k,h,\text{now}} - t_{i,j,h,\text{now}}|$. This penalty is our way to model the error due to the asynchronicity of the two FX quotes used to compute $x_{i,j,k,\text{vehicle}\,h}$. This seems to be a fair assumption, but a closer analysis might lead to a slightly modified choice of the penalty.

The error due to the age is modeled as $c_A\sqrt{A}$, thus using a scaling law. In reality, this scaling law also depends on the volatility of the FX rate whose seasonal behavior can be captured through ϑ-time. In our algorithm, we neither analyze the volatilities of the different time series nor use ϑ-time, but we simply choose a constant c_A. This is justified by the fact that the age of quotes only affects one ingredient of the error, often a less important ingredient, given the high frequency of the data. However, in special situations such as weekends, a more sophisticated modeling approach may become desirable in the future. In our model, a reasonable choice is $c_A \approx 0.0002$ where time is expressed in minutes. This roughly corresponds to an annualized volatility of 14%. While typical values of annualized volatility are just around 10%, we choose a value that reflects the situation of an elevated volatility.

The bid-ask spread also defines an uncertainty of the price level which constitutes yet another ingredient of the error. We simply define this error as $0.4s$, slightly less than half the relative spread.

Different value dates of underlying FX rates also lead to an error of computed cross rates. Instead of launching a tedious research study on local holidays, value date rules and interest rate differentials, we assume a very small, constant error ingredient of size $E_{\text{valueDate}}$ for all computed cross rates, irrespective of FX rate and date. A reasonable choice is $E_{\text{valueDate}} = 0.00004$.

There is no reason to assume any particular correlation between the different error ingredients. Therefore the squared total error can be modeled as the sum of all squared error ingredients. For direct quotes, we obtain

$$(\delta x_{i,j,k})^2 \;=\; b^2_{i,j,k} + c^2_A\, A_{i,j,k} + 0.16\, s^2_{i,j,k} \tag{3.8}$$

and, for computed cross rates,

$$(\delta x_{i,j,k,\text{vehicle}\,h})^2 \;=\; b^2_{i,j,h} + b^2_{i,k,h} + c^2_A\, A_{i,j,k,\text{vehicle}\,h} + 0.16\, s^2_{i,j,k,\text{vehicle}\,h} + E^2_{\text{valueDate}} \tag{3.9}$$

Thus the total error of computed cross rates distinctly exceeds that of direct quotes, in normal cases.

The total error determines the impact of a quote in the multivariate pricing engine. The smaller the error, the higher the weight.

3.4 Determination of the most correct prices

In the previous parts of Section 3, we have done the necessary preparation of the multivariate algorithm. Different FX streams are synchronized, cross rates can be computed, and the expected errors of directly quoted and computed FX rates can be assessed.

Now we can tackle the multivariate FX rate correction which is the heart of the algorithm. The corrected FX prices are weighted averages of raw prices and prices computed through all possible vehicle currencies. These averages should be *robust* averages. This is necessary because the error distribution of prices is not normal. Conventional averages, weighted by inverse squared errors, would be perfect for normally distributed errors[1]. The error distributions are leptokurtic, though. In some cases such as rapid price moves and numerical errors by data contributors, some outlier values can strongly exceed the expected mean error (and still be moderate enough that the underlying raw prices pass the univariate data cleaning filter). An example is a computed cross rate where one of the underlying raw FX rates is stale or of generally low quality during some periods. The outliers would have a too high weight in a conventional average.

The classical choice of a robust mean is the *median*. The median often implies selecting a reasonable price rather than averaging between all available prices. Weighting is still essential, so we use a *weighted median*. In normal situations, the direct quote from the best data source has such a high weight that its value often coincides with the weighted median. Then the output series looks identical to the input series of the best source. If the best source has a data gap, the weight of the last direct quote will decline with time, so the weights of the computed cross rates or the weights of another source will gradually take over.

The weighted median is defined and computed as follows. First, we *sort* all the N available prices for the currency pair j,k, $x_{i,j,k}$ as well as $x_{i,j,k,\mathrm{vehicle}\,h}$, according to their values. All available data sources and all available vehicle currencies are used. Note that we can add a stabilizing but conservative and sometimes delaying element by including the *previously* computed, final x value of the same currency pair in the median calculation. This is not really recommended, but mentioned as a possible stabilizing element. The new ordering of the sorted prices is denoted by the new index l: $x_{j,k,0} \leq \ldots \leq x_{j,k,l} \leq \ldots \leq x_{j,k,N-1}$.

The choice of the weights requires a special discussion. For a weighted arithmetic mean, the natural choice would be the inverse squared errors. If the errors were normally distributed, this choice would lead to the maximum-likelihood estimate of the true price. Unfortunately, the error distribution is leptokurtic. In some very rare situations, even the best source can be bad – much worse than expected under a normal distribution. This is why we choose a robust mean, the median. If we use inverse squared errors as weights of the median, the best source will determine the resulting median as long as its quotes are fresh. In other words, the median is not robust in the very rare case of outliers in the best source.

A weighted median can be made more robust by equalizing the weights. This means higher weights for bad sources and lower weights for good sources. It goes without saying that such an equalizing correction is dangerous. It should not go as far as to give equal weights to all observations. The choice of the weight equalization formula implies a trade-off. Equalizing makes the results more robust, at the cost of fast reaction when prices move rapidly. In the case of sudden trends, the best source is likely to move first. By reducing the weight of this best source, the resulting data will reflect the sudden trend with a small delay. By examining several difficult situations of different kinds, Kris Meissner and Ulrich Müller have found a suitable trade-off by using the cube root of the inverse squared errors as the weights:

$$w_{j,k,l} \;=\; \left(\frac{1}{(\delta x_{i,j,k})^2}\right)^{\frac{1}{3}} \;=\; \frac{1}{(\delta x_{i,j,k})^{2/3}} \tag{3.10}$$

[1]This rule can be found in standard textbooks and ultimately originates from maximum likelihood estimation.

and, for computed quotes,

$$w_{j,k,l} \;=\; \left(\frac{1}{(\delta x_{i,j,k,\mathrm{vehicle}\,h})^2}\right)^{\frac{1}{3}} \;=\; \frac{1}{(\delta x_{i,j,k,\mathrm{vehicle}\,h})^{2/3}} \tag{3.11}$$

If some initial numbers are very different, their cube roots are closer to each other. This is the equalizing effect. Using the square root implies weaker equalizing, the fourth root means stronger equalizing.

The median is found at the index $l = L$ that occupies the mid-point of the cumulated weight:

$$L \;=\; \min\left\{ l' \;\middle|\; \sum_{l=0}^{l'} w_{j,k,l} \;\geq\; 0.5 \sum_{l=0}^{N-1} w_{j,k,l} \right\} \tag{3.12}$$

The resulting median is

$$\bar{x}_{j,k} \;=\; \begin{cases} x_{i,j,L} & \text{if } \sum_{l=0}^{L} w_{j,k,l} \;>\; 0.5 \sum_{l=0}^{N-1} w_{j,k,l} \\ 0.5\,(x_{i,j,L} + x_{i,j,L+1}) & \text{if } \sum_{l=0}^{L} w_{j,k,l} \;=\; 0.5 \sum_{l=0}^{N-1} w_{j,k,l} \end{cases} \tag{3.13}$$

The subscript “now” generally applies to the variables but has been dropped to avoid a too heavy notation.

3.5 The resulting error

The mean logarithmic price of Equation 3.13 is an approximation of the true logarithmic price, but still has a certain error. If the mean logarithmic price was an arithmetic mean of normally distributed observations, the squared error would be computed according to the following textbook formula:

$$(\delta\bar{x}'_{j,k})^2 \;=\; \frac{1}{\sum_{i=0}^{n-1}\left[\delta_{i,j,k}\,\frac{1}{(\delta x_{i,j,k})^2} + \sum_{h=0}^{m-1} \delta_{i,j,h}\;\delta_{i,k,h}\;\frac{1}{(\delta x_{i,j,k,\mathrm{vehicle}\,h})^2}\right]} \tag{3.14}$$

$$=\; \frac{1}{\sum_{l=0}^{N-1}\frac{1}{(\delta x_{j,k,l})^2}} \;=\; \frac{1}{\sum_{l=0}^{N-1} w_{j,k,l}}$$

where the last two forms make use of the sorted sequence of price observations and their errors. The quantity $\delta\bar{x}'_{j,k}$ is not yet the final form of the error, hence the prime symbol.

The median of Equation 3.13 is not a conventional mean, so there is no theory that could prove that Equation 3.14 is the correct expectation of the squared error of the median. Equation 3.14 simply provides our approximate assumption on the error. The usefulness of this assumption will be determined in practice. One modification for practical reasons already follows. In order to avoid an underestimation of the error, we force a low minimum value:

$$\delta\bar{x}_{j,k} \;=\; \max[\delta\bar{x}'_{j,k},\; b_{0,j,k}] \tag{3.15}$$

The minimum is the basic $b_{0,j,k}$ error assumed for the best source, which has the index 0.

Why do we care about the error of $\bar{x}_{j,k}$? The pricing engine has to supply corrected bid-ask prices, not only middle prices. The resulting error $\delta\bar{x}_{j,k}$ of Equation 3.15 will be used to determine the resulting bid-ask spread, as to be shown in Section 4.

4 Production of output data streams

4.1 Generating bid-ask spreads

In Section 3, we have obtained a resulting logarithmic middle price $\overline{x}_{j,k}$ and its expected error $\delta\overline{x}_{j,k}$, but we still need a bid-ask spread. In Section 3.3, we have modeled the error due to the bid-ask spread to $0.4s$. Now we invert this relation to generate a bid-ask spread from the error. The inverted relation yields a spread of 2.5 times the error:

$$\overline{s}_{j,k} = 2.5\,\delta\overline{x}_{j,k} \tag{4.16}$$

This is the bid-ask spread to be used in the output price calculation.

4.2 Timing of the output

The multivariate price correction algorithm of Section 3 is used to produce output data streams of corrected data. The first idea is to run this algorithm for each new incoming tick of any of the input time series. Each such tick defines a new t_{now}. Each new multivariate computation results in a full set of new quotes for all FX rates that can be constructed from the pool of currencies.

If this first idea is implemented, we obtain data streams of extremely high frequency. The resulting frequency of each output series will be equal to the frequency of all input series *together*. There will be many ticks per *second* during working days. This enormous data frequency will require a huge amount of storage space, CPU time and transmission bandwidth. On the other hand, most of the tick innovations in these highest-frequency series will be trivial. New ticks will often deviate from predecessor ticks by small fractions of a basis point. Users do not want very frequent updates with trivial information.

In this situation, we introduce rules that cut down the output frequency. The guideline is to produce a corrected quote only if it contains non-trivial information. This can be reached by the following measures:

- Cutting the frequency of multivariate computations. A multivariate computation is done only if the last such computation happened more than, say, 20 seconds ago or if the new tick value $x_{i,j,k,\mathrm{now}}$ deviates from the predecessor value by more than 0.00005. Of course, these criteria can be adapted to the user's needs.

- Cutting the output frequency of individual FX rates. We store the old logarithmic bid and ask prices of the output, i.e. $\overline{x}_{j,k} - 0.5\overline{s}_{j,k}$ and $\overline{x}_{j,k} + 0.5\overline{s}_{j,k}$. If one of the newly computed bid or ask values deviates from the predecessor value by more than 0.0001 or the last output has a certain minimum age, we produce an output tick. Otherwise, the old output tick can be regarded as valid, and no new tick is produced.

- Producing output data only for those cross rates that are really needed by the application.

The exact choice and fine-tuning of these rules has to be done in the test phase and in practice. When applying the rules, we obtain output streams of data that are perceived as being reasonable, as every tick conveys some non-trivial information.

4.3 Output prices

The logarithmic prices obtained so far have to be transformed to nominal prices in the usual output format:

$$p_{\mathrm{corr,bid},j,k} = e^{\bar{x}_{j,k} - 0.5\bar{s}j,k} \tag{4.17}$$

$$p_{\mathrm{corr,ask},j,k} = e^{\bar{x}_{j,k} + 0.5\bar{s}j,k} \tag{4.18}$$

The precision of these prices needs some discussion. Practitioners like to have prices rounded to a particular precision, where the last digit of the resulting numbers defines the unit “basis points”, “pips” or “tics” (where the latter word may lead to confusions with our usual word “tick” for data records). The correct number D of digits after the decimal point as used by practitioners can be configured by hand. There is a also a rule of thumb to determine the correct number of digits:

$$D = 5 - \mathrm{ceiling}\left(\frac{\bar{x}_{j,k}}{\log 10} + 0.4\right) \tag{4.19}$$

where log means the natural logarithm and the function ceiling returns the next higher or equal integer number. For major rates, this rule is correct, but practitioners should check the results and confirm that this rule produces the right number of digits also for minor FX rates. For some exotic FX rates with unusual numerical values and for regulated (not freely floating) FX rates, the rule may be wrong.

5 Conclusion

The algorithm proposed in this document is able to fulfill its task. The first results of a prototype and the FXBlender demonstrate the successful generation of output streams. In some known cases where the raw ticks had a bad behavior, the output ticks are of high quality.

At some points, the algorithm can be enhanced. The growth of the data error with increasing quote age is modeled with a very simple scaling law that can be improved by a sophisticated analysis of volatility and the introduction of ϑ-time. This will probably improve the behavior of the pricing engine especially over weekends and after data gaps.

The presented methodology also bears some promise for generating quality data for other financial instruments. All interest-rate related instruments, for example, could be pooled together, leading to a set of instruments similar to the multivariate set of FX rates treated in this document. An analysis of yield curves and correlations would play a role analogous to the analysis of triangular arbitrage between FX rates.