

Visit **Benefits and Pensions Monitor** magazine online at: <http://www.bpmmagazine.com>

Fixed Income Attribution With Minimum Raw Material

By: Andrew Colin

Security level approaches to fixed income attribution currently require all the information necessary to price an instrument from first principles or an external source of risk sensitivity measures. Both techniques require a supply of data for which a performance calculation oriented back office will probably not have access.

For many organizations, the sourcing and integration of this new data is one of the principle obstacles to implementing a detailed fixed income attribution capability.

However, by using recent advances in machine learning techniques, it is possible to obtain highly detailed attribution analyses with much less user-supplied information than has previously been the case.

Attribution On A Single Bond

The common feature between all fixed income attribution schemes is that they provide a link between sources of risk and performance. For instance, the main source of performance-generating risk for a vanilla corporate bond is just the bond's yield to maturity – a single proxy measure that represents the current term structure.

Yield is a scalar-valued measure that can be decomposed into Treasury curve levels, credit spreads, and other effects such as liquidity, prepayment, and market noise. If, for example, credit spreads contract, the overall yield of the bond will fall, its price will rise, and extra performance will be generated by the exposure to credit spreads.

By measuring yield changes from various sources, the performance due to each source of risk may be measured. This return can then be aggregated over the entire portfolio and compared to the corresponding return from the benchmark. In this way, the skills of the portfolio manager at predicting and managing various types of exogenous risk can be measured and assessed.

Linking Risk To Return

To measure the effect of a change in yield on a bond's price, we can either price the bond using a standard pricing formula or we can use some measure of price-interest rate sensitivity such as modified duration. These typically give results that are very close, but they both have well-documented data issues.

To calculate attribution returns, the change in yield is decomposed into a number of sub-changes such as treasury curve shift or credit spreads. This is known as first-principle pricing. The price of the security is calculated using the yield at the start of the calculation interval and then recalculated after including each successive yield change. The differences between the resulting prices can then be used to calculate the return arising from each source of risk.

Pricing of a security from first principles requires that we actually know a suitable pricing algorithm and the market's pricing conventions. In addition, first-principle pricing requires that we have available the coupon, maturity date, coupon frequency, and other security-specific information for the bond. Obtaining this data is not easy, particularly where large benchmarks with many thousands of bonds are to be modeled, or for floating-rate notes where coupons may be reset on a monthly basis.

An alternative approach is to form a Taylor expansion for the price of the security. This provides a convenient and rapid means to calculate the effect of various changes in yield on performance as no pricing machinery is involved. However, a source of risk data is

required, which has to be sourced externally. In addition, the implied linear relationship between pricing inputs may not always hold, in which case the accuracy of the analysis is likely to be poor.

Reduced to its simplest form, the common feature of both of the above approaches – and the main requirement of an attribution analysis – is simply a function that maps changes in yield (and other market-imposed quantities) to performance. This leads us to a third attribution approach.

By using machine intelligence algorithms, we can train a universal function approximator to learn the relationship between yield and price, without having to specify the form of this relationship. The algorithm considered is a neural network.

Neural Networks

We define a neural network as an array of connected nodes. Nodes are grouped into separate layers and a network is organized as an input layer, one or more hidden layers, and an output layer. Each node in a given layer is connected to every other node in the next layer via a link, and each link has an associated weight. The information in each layer can be mapped to an 'm' (the number of nodes in the starting layer) and 'n' (the number in the ending layer) matrix. Every node has a scalar-valued activation level.

Suppose that each node in the input layer is activated to an externally set value. For instance, we might set the first and last nodes in the input layer to a level of '1' and all the other nodes to a level of '-1.' The signal is propagated through the network feeding the resulting signal from the hidden layer to the output layer so that the initial signal cascades through the network. This is an example of a 'feed-forward' network. At the end of the 'feedforward' process, the output node has an activation level that represents the network's response to the initial activation vector.

Various robust algorithms exist to train a neural network to derive a set of weights to match a set of given input and output vectors. The best-known training algorithm is the 'back-propagation of errors' technique. Other techniques include simulated annealing, genetic algorithms, and particle swarm optimization.

Whichever algorithm is used, the ideal outcome is for the network to learn all of the data presented, so that if it is subsequently presented with one of the training samples after training is complete, it will produce the same output vector that was originally presented in training. We say that a network has converged if the values of the output vectors, produced by some training algorithm, are close to the corresponding true values.

Convergence may not be achieved if:

- there is little or no structure in the training data, in which case there is no implicit model to learn
- there is a great deal of noise in the training data
- the network topology has insufficient flexibility to learn the details of a complex relationship.

The first two are unlikely to be the case when training a network to price a fixed income security and the last can be addressed by adding more capacity to the network in the form of extra nodes in the hidden layer.

For attribution, a neural network has two particularly important and desirable features:

The first is that neural networks act as model-free function estimators. In other words, given a set of input and output data, it is not necessary to know very much about the internal function being modeled in order to train the network and to model the relationship implied by the supplied data. This makes the use of neural networks particularly suitable for approximate security pricing, where we know the values of the pricing inputs (such as yield, repayment rates, or interest rate volatilities) and the outputs (the market price and returns for the security), but the relationship between the two may not be fully specified analytically.

The second is that a 'feed-forward' network will also interpolate data in a sensible way. Unlike a high-degree polynomial, which may model a data set exactly at a set of samples, but whose interpolated values may vary widely from the underlying data set, the way that a 'feedforward' network is designed means that input vectors that do not precisely match an existing input will still result in sensible outputs. The network will use the knowledge abstracted from the various training cases and provide a weighted average of the various cases that is the closest match to the new data.

Learning To Price A U.S. Treasury Bond

To demonstrate the technique, we calculated the capital price of a U.S. Treasury bond with coupon and yield varying randomly between two per cent and seven per cent, a maturity date of January 1, 2050, and settlement dates varying randomly between January 1, 2000, and January 1, 2050. The long maturity date was chosen to ensure that the bond showed high convexity and that the yield-price relationship would be non-linear. Five hundred prices were calculated and used as training data and a further 500 prices were then calculated by the network on previously unseen data samples. The selected inputs were settlement date, coupon, yield, and maturity date and the output was the bond's capital price, quoted using the standard convention per \$100 face value, just as for a standard pricing formula. Figure 1 shows the distribution of forecast prices against true prices.

There are several points to note from these results.

The accuracy of the learned results is very high, even on previously unseen data. It appears that the neural network can indeed learn the underlying form of the equation that relates yield to price, without the user having to specify the form of this relation in any way.

The accuracy of the results is not high enough for exact pricing. For instance, one could not use neural network-based pricing to produce deal tickets, where exact agreement on prices between counterparties is vital. For this reason, deal prices are usually calculated using standard valuation formulae with highly detailed documentation on usage and rounding conventions.

However, we contend that this is unimportant for attribution. In attribution analysis, we are primarily concerned with achieving a qualitative understanding of the relationships between risk and rewards in a managed portfolio and, for this purpose, a discrepancy of a fraction of a fraction of a percentage point between risk sources is unlikely to be important, particularly when there is wide variation in the techniques used to decompose yield curve movement types.

The main benefit of using neural network based attribution is that it is possible to dispense with much of the data required by both the first-principle and perturbational approaches.

Firstly, the precise form of the pricing equation need not be specified. From the network's point of view, the relationship is already specified in the data. The situation is analogous to teaching a child to catch a ball – the relationships involving perspective, physics, and gravity being required.

The second benefit of using machine intelligence techniques for attribution is that we can exchange many of the data maintenance requirements in more traditional approaches for number-crunching power. As the results show, the less data supplied, the higher the computational resources required.

Neural network-based pricing offers a powerful tool for fixed income attribution. Its ability to model pricing relationships without any user-supplied intervention, solves one of the major operational difficulties in the implementation of an attribution system – that of accurately modeling the relationship between risk and performance at a security-level basis. ■

Andrew Colin is with StatPro Australia.