How would an historian do time-series econometrics? Analogy weighting using machine learning

Jim Savage

7 July 2015

Abstract

Introduction

In building reduced-form empirical macroeconomic models for policy simulation and forecasting *today*, how much history should be included? Too much history and the analyst risks estimating parameters that describe history well but have little relevance to today. With too little history, estimates risk missing out on useful information. Implicit in doing time-series work is an assumption that history provides a useful guide for parameterising economic models.

The method presented in this paper takes this one step further: I assert that if history provides a useful guide to reduced-form models, then more analogous histories should provide a better guide. Conversely, we should have little confidence in model projections being made if the current state of the world is markedly different from the world from which the model parameters were estimated.

How should we decide which histories are most relevant to today? The techniques most commonly used to address this problem aim to account for unmodelled structural change in the underlying data generating process. They come in varying degrees of complexity.

A naive or un-principled approach is to assume that co-movements between variables over the recent history are likely to emerge from a structural relationship that is relevant for the near future. Making such an assumption motivates techniques often used in industry and government work[cite: BASEL III], such as estimating models over a rolling window, or using exponential smoothing to give less weight to more distant historial observations. These techniques are easy to understand and implement, but it is not clear what an optimal window selection rule should be.

A second class of strategies includes all historical data, but explicitly models the time variation in model parameters, either using regime-switching frameworks or continuous-state approaches. These approaches are more principled, in that the parameters and data come from a generative model (requiring that the researcher be explicit in their beliefs). Yet they are lacking on other dimensions. The models are typically difficult to implement, are richly parameterised (necessitating informative priors), and, often, the generative models themselves give non-zero probability to implausible outcomes, such as explosive states.

How would historians build econometric models? The 'historianist' approach proposed in this paper is to analogise today with periods of history that have similar characteristics, and draw inference based on those analogies. There is a large problem in applying this historianist approach formally to time-series models: how can we determine which time-periods are "similar" to today? Estimating similarity typically involves selecting a metric with which we can measure the similarity between periods. But how much weight should be given to the various dimensions? And should these weights change over time? Perhaps one variable is very important in some states of the world and not in others.

This paper describes an algorithm that makes use of the Random Forest (Brieman, 2001), a popular method from machine learning, to construct "analogy weights". These weights can be used to help regression models learn more from similar periods of history. In the applications below, analogy weighted models outperform benchmark models in out-of-sample predictive comparisons. Yet the real advantage of using analogy weights is to inform the user when the world is in an unprecedented state. When used with Bayesian regression models with informative priors on (large) residual variances, this algorithm can help increase uncertainty

when the model is being asked to generate projections in a world that is not similar to the world used to estimate the model.

The weights have two important features. First, their construction does not make use of a metric; a variable's importance to the weight can vary with states of the world. Second, there are automated out-of-sample robustness checks in the construction of the weights, and so weights represent only similarities on dimensions that appear to have a robust relationship to the dependent variable. The algorithm also requires very few tuning parameters, making it a useful turn-key solution to many time-series modelling tasks.

The paper also presents two applations of the algorithm. The first is in estimating "core" inflation—the permanent component of inflation, using a Vector Autoregression (VAR) with analogy weights. The second is an application to portfolio risk management, in particular estimating a conditional volility model for a small portfolio of stocks during the Global Financial Crisis. This second application illustrates how researchers can easily incorporate analogy weights into multivatiate GARCH models (below, my own augmentation of the CCC-GARCH method of Bollerslev (1990)). The applications show that using analogy scores generates the greatest benefit in the first few periods following an unprecedented regime change.

1: Historical analogies in the literature

The premise of analogy weighting is that periods of history that are similar to today may provide more information about the structure of contemporaneous economic relationships than less similar periods of history. This is not a new revelation—many theoretical models allow similar states of the world to emerge. Consequently, many empirical models explicitly model regime shifts. It is also not a revelation only relevant to econometrics. In the finance, machine learning and signal processing literature, similarities between time series are often used in predictive tasks.

Why may similar periods be more relevant to today than dissimilar periods? This is a largely theoretical question left for further research, yet the literature contains many appealing

models that give some support to the notion that the behaviour of the economy may depend on the values of macroeconomic and market variables, particularly during crises. The most prominant example of this is in Kiyotaki and Moore (1997) in which exogenous technology shocks reduce the economic value of capital and so affect asset prices. Faced with binding collateral constraints, firms cease to be able to finance investment, and so reduce their asset purchases; this reduces asset prices and amplifies the cycle. Bianchi (2010) proposes a similar mechanism, in which borrowers of foreign currency face similar collateral constraints when their home currency depreciates against the creditor's, reducing the availability of foreign credit and thus making exchange rate depreciations contractionary. In Brunnermeier and Sannikov (2013), "expert" firms can absorb small shocks by adjusting payouts and re-building capital, but under large or serially correlated shocks these firms may be forced to stop investing or liquidate their capital. Consequently, the impact of a shock in their model depends on the sign and magnitude of the shock, and the economy's distance from its steady state.

While the channels by which these models transmit shocks vary, their implications for reduced-form modelling are similar: the behaviour of economic agents probably varies across states of the world, and there are potentially many such states.

Regime-switching models as in Hamilton (1988, 1989) and as reviewed in Kuan (2002) and Hamilton and Raj (2013) attempt to model changes in the state of the economy by allowing parameters to vary across a discrete number of regimes. These models have been applied in a variety of settings, such as examining fiscal multipliers in recessions vs expansions (Auerbach and Gorodnichenko 2012) and the permanent and transitory impacts of business cycles across different sub-periods of the business cycle (Kim and Murray, 2001). Another interesting application to currency volatility is in Klaassen (2001). In that paper, standard GARCH forecasts of volatility tend to be too high during high-volatility periods; this is because long periods of relative calm result in a GARCH model's persistence parameter being very high. Allowing GARCH parameters to vary across states results in better out-of-sample volatility forecasts.

The basic structure adopted by the regime-switching literature is to allow for some model

to depend on parameters that vary across $s \in S$ different states: $y = f(\theta_s, x)$. The states themselves are not known, but need to be inferred based on observations of y. Of central interest is the probability of transitioning from state j to state k $p(s_t = k|s_{t-1} = j, \ldots, s_1 = l, y_{t-1}, \ldots, y_1)$. If this probability is equivalent to $p(s_t = k|s_{t-1} = j)$, then the state transitions according to a Markov Chain.

Regime switching models are clearly a powerful approach at capturing un-modelled variation across time, especially if the differences between parameters' values in one state and another are large (this scenario may be difficult to model in the continuous state approach below). Yet the main drawback is that the approach relies on all regimes having been observed in the historical series. This limitation reveals itself in overfitting (in which out of sample performance is significantly worse than in-sample performance). This anecdote, from Fabozzi et al (2011) illustrates the point well:

> A survey participant at a firm where regime-shifting models have been experimented with commented,
>
> > Everyone knows that returns are conditioned by market regimes, but the potential for overfitting when implementing regime-switching models is great. If you could go back with fifty years of data—but we have only some ten years of data and this is not enough to build a decent model.

As the number of potential states of the economy grows, it may make more sense to use continuous-state models, as described in Koop and Korobilis (2013). In this approach, rather than having a set of parameters for each state of the economy, the "state" of the economy *is* the set of parameters; these are allowed to vary over time. A common implementation of these models, as in Cogley and Sargent (2001, 2005) and Koop et al (2009), Koop and Korobilis (2009, 2013) models parameters as changing over time in a random walk process, rather than modelling parameter values as responding directly to changes in exogenous variables. These papers implement some version of the following model. Taking some vector of variables y, they estimate the state space vector autoregression

$$y_t = Z_t \beta_t + \epsilon_t$$

Where Z_t contains an intercept and p lags of the dependent variables. The state evolves according to a random walk

$$\beta_t = \beta_{t-1} + \eta_t$$

and the innovations to both the VAR and state processes are multivariate normal. In some implementations (such as Cogley and Sargent's 2005 model) the covariance of VAR innovations is itself time-varying, in which case these form a part of the evolving state.

$$\epsilon \sim \mathcal{MVN}(0, \Sigma_{(t)})$$

$$\eta \sim \mathcal{MVN}(0, Q)$$

Given priors on β_0, Σ and Q, the papers above estimate the time-varying β using a Kalman Filter. One common modification (as in Koop and Korobilis 2013 and Raftery et al 2010) is to include in the filter a "forgetting factor"—a parameter that determines how much past information influences the current estimate of β_t. The implicit assumption made when using this approach is that the distant past is irrelevant to parameter estimates today.

Both methods described above—regime shifting models and continuous-state-space models—offer some real benefits to a researcher modelling unobserved structural changes. Yet the models have drawbacks. Like typical reduced-form models, discrete regime-switching models are unable to generate states that have not been observed. If the true data generating process has not revealed all her states, these methods are prone to inductive errors (Popper, 1934). In contrast, continuous-state models give positive weight to the possibility of as-yet unobserved states. But they are hard to implement, and tend to be very richly parameterised. A large number of parameters relative to the number of observations requires that informative priors

must be used to estimate continuous-state models (see Koop and Korobilis, 2009).

These problems are not unique to econometrics. In the machine learning and signal processing literature, "time-series-clustering" techniques aim to group time-series together into clusters, each of which should contain time-series that behave similarly. When researchers are looking for analogous periods of a given time-series, they use "subsequence clustering", in which a rolling window generates many time-series which can be clustered.

One approach (summarised in Keogh and Lin, 2003) has been to compute the values of a point-wise distance (usually a L1 or L2 norm) between all adjacent points in two time-series, and summarise the "similarity" between the two series with the average value of the distance. Time series are then clustered using K-means or hierarchical clustering. This approach is extremely computationally expensive when the length of time-series is large, and impossible when there are missing values or time-series are different lengths. In Keogh and Lin (2003), this method is shown in many cases to generate "meaningless" clusters—clusters of time series that perform for prediction no better than random groupings.

Another approach has been to summarise time-series in terms of their "characteristics", and then cluster on these. This is less computationally draining, and allows the researcher to cluster series that have different lengths, missing data, etc. In Deng et al (1997), each series is characterised by the parameters of an ARMA model fitted to it, and series are clustered based on these parameter estimates. In Wang and Hyndman (2006), each time-series is summarised according to characteristics including frequency, serial correlation, trend, skewness, kurtosis, seasonality, etc., and series are clustered based on the values of these characteristics. This approach to time-series clustering has been more successful at generating useful clusters.

There are a few aspects of time-series clustering approaches that make them unappealing for applied time-series economics research. First, clustering methods are used to cluster *univariate* time series with other univariate time series. The economic approaches discussed above aim instead to generate similar *histories*, which may be defined by the values of many time series. Second, while clustering may make sense if univariate time series behave in a deterministic manner, many interesting economic time series (including exchange rates, stock prices and consumption) look like random-walk series in the short-run. As in Keogh and Lim

(2003), clustering random walk series predictably results in spurious clusters. Finally, most applied economic work is concerned with causal analysis, not prediction. Even if building univariate clusters was able to improve prediction of economic time series, it is not clear why this should help in building causal models.

How this paper relates to the literature

The algorithm described below aims to overcome the difficulties associated with both regime-switching models and continuous-state models, while retaining some of the advantages of both. Like regime-switching, the model allows large jumps in the values of parameters as the state shifts. Like a continuous state model, parameters are continuous across states of the world (including as-yet unseen states). The big advantage is the ease of application. While the existing methods require a great deal of expertise, time and highly informative priors, analogy weighting can be as simple as using (analogy) weighted least squares. The cost of the approach is that we move away from a generative model.

The method works by building a weighting vector that relates one period in history (normally today) with all other periods in observed history. This score serves as a "distance between histories", and is defined on the characteristics of the economy or system at that point in time.

To avoid having to define a metric that measures similarity between periods of history, the algorithm proposed makes use of the Random Forest (due to Brieman 2001), a tool from machine learning. A biproduct of fitting a Random Forest is a proximity matrix, which for all pairwise observations provides a measure of proximity in the Random Forest space. The main benefit of this tool is that the distance measure is the distance between observations *in terms of the variables that matter to the prediction of the outcome variable, and the importance of each variable can vary over time.* The algorithm is a time-series generalisation of the approach taken by Miller Savage and Tan (forthcoming), in which Random Forest proximities are used to construct a matching estimator that is less sensitive to the Smith and Todd critique than is Propensity Score Matching.

Introducing Random Forests

The algorithm described below uses the proximity matrix of a random forest. This chapter describes how the Random Forest works, starting by explaining the Classification and Regression Tree method, and then describes how the proximity matrix is estimated. Readers with a familiarity with these methods may skip to the next chapter.

Regression Trees

Classification and Regression Trees, known as CART and due to Brieman et al (1984), is a non-linear supervised recursive partitioning method. While this technique is not widely used in economics, it has recently found interest among some prominant researchers, such as Athey and Imbens (2015) and Varian (2014). The aim of the technique is to partition a dataset according to the values of its covariates such that in each of the sub-groups the values of the dependent variable are more similar than in the parent group. The routine described below is a simplified version of the CART routine implemented in many software packages, though one that is much closer to the trees that make up a Random Forest.

Formally, let $x_{i,p}$ be the $p \in P$-th covariate for individual $i \in I$. Although in some implementations of CART x can be a categorical variable, we consider only the case with a numerical x. The collection of all xs is X. Each individual has a dependent variable $y_i \in Y$, which again we take to be numerical.

Each node of the tree—a juncture between branches—is associated with a "split-point" $\bar{x_p}$ defined as the point that splits the parent group into two child groups so that the sum of sum of squares for the child groups is reduced by as much as possible:

$$\bar{x_p} = \text{argmax}_{x_p} \left\{ \sum_{i \in I} (y_i - \bar{y})^2 - \left(\sum_{i \in I_L} (y_i - \bar{y_L})^2 + \sum_{i \in I_R} (y_i - \bar{y_R})^2 \right) \right\}$$

Where I_L is the "left" part of the data; the individuals i for whom $x_{i,p} < \bar{x_p}$. The mean of those individuals' dependent variable is $\bar{y_L}$. I_R is the remainder of the observations. These split points are simply values of the covariates that appear to be useful in terms of splitting

the group into more similar sub-groups.

CART is a recursive algorithm, in that once a split has been estimated, the two sub-groups are then split, recursively, until further splits result in child groups groups that are no more similar than their parents, or until a minimum node size has been reached. In "pruned" tree, there is some cost attached to making further splits; if the gain in terms of decreased sum of squares does not meet some cost, then the split is not made. This is an efficient way of decreasing over-fitting.

Once a tree is estimated, it can produce predicted values for a new observation *even if that observation is missing elements of* X. A new observation is simply "dropped" down the tree until it either reaches a terminal node (one with no child nodes) or until it reaches a node whose split is defined on a variable that is missing for the individual. Once it reaches this node, its predicted value for y is simply the average y for observations in that node in the training set.

If two observations reach the same terminal node, they are said to be *proximate*. Two proximate observations are similar in terms of all variables on which splitting decisions are made.

Random Forests

While CART models are extremely easy to understand and visualise, they do tend to result in overfitting. They also do not capture smooth relationships well (there will only be as many distinct predicted values as there are terminal nodes).

Random Forests (Brieman, 2001) were designed to address these problems. They are essentially a collection of (normally thousands of) CART models, with two small twists. First, in a Random Forest, each tree is estimated on a random subset of the observations I. Second, at each node, only a proportion of all candidate x_p are eligible to be split on. All trees are grown to their full extent—that is, not pruned. This results in the trees that make up a Random Forest being quite distinct, there being no individual observation or x variable that has an enormous influence over all trees as a group.

To generate predicted values from a Random Forest, a new observation x_i is passed down all trees, and the fitted value is the average of the fitted values in all trees.

Random Forest proximity matrices

This paper is not concerned with the (impressive, see Caruana and Niculescu-Mizil, 2006) predictive abilities of the Random Forest. Rather, the Random Forest generates a "proximity matrix" that gives a distance between all observations in the dataset.

When two observations fall into the same terminal node as one another, they are said to be *proximate*. The (i,j)th entry in a Random Forest's proximity matrix is the proportion of terminal nodes that observations i and j share across all trees. For additional robustness, it is desirable to use only "out-of-bag" observations in the calculation of proximities. These are the observations excluded when each tree is grown. In this formulation of the proximity matrix, the proximity score is the proporton of terminal nodes shared by two observations *for the trees that the observations were out-of-bag*.

The proximity matrix is an attractive candidate for a distance measure between historical observations. It does not require defining a distance metric or estimating importance weights for the covariates. Because the Random Forest is supervised (it has a dependent variable), the proximities can be thought of as the distance between observations *in terms of the variables that matter for the prediction of y*. And because of the robust manner in which a Random Forest is estimated, (stationary) independent variables whose relationship with y is coincidental are less likely to influence the proximity score. Consequently, the analyst can afford to include many independent variables; those without a robust relationship will tend to be excluded.

Analogy score weighting—the algorithm

The aim of the following algorithm is to generate a vector corresponding to periods $t_1, \ldots, T$ that contains weights denoting the similarity between all historical periods and period T.

Period T will typically be chosen to be the most recent period, as the model builder will generally want to build a useful model of the current state of the economy.

The first step is to construct a $T \times P$ matrix X that contains P variables that could be though of as describing the "state" of the economy/market conditions. These variables should be stationary.

An aggregated proximity matrix is then constructed like so:

```
initialise pmat as a T by T matrix of 0s


for p in 1 to P:
  build Random Forest model of X[,p] | X[,-p], call this RF
  pmat = pmat + proximity matrix of RF
end
pmat = 1/P * pmat
```

The T-th row of this matrix contains the proximities between period T and all periods beforehand. This vector can be then used for weighting observations in a regression model.

Tuning parameters

Given the introduction of randomness in estimating the random forests that we use to estimate analogy weights, how sure should we be that a given proximity is meaningful? That is, is there a sense of "significance" of the proximity estimate?

Two of the random forest's tuning parameters have the greatest influence on proximity estimates. The first is `ntree`, the number of trees in the forest. A smaller number of trees relative to the number of observations is more likely to generate spurious matches. As the number of trees grows, the stability of proximity weights improves. This can be seen by taking the absolute value of the derivative of average proximity scores with respect to `ntree`. That is:

$$\text{instability score} = \text{abs}\left(\frac{\partial \frac{1}{n^2}\sum_i \sum_j \text{proximity}_{i,j}}{\partial \text{ntree}}\right)$$

This is illustrated below. As is clear, there are diminishing returns to the number of trees.

The second main tuning parameter that users should be aware of is the size of terminal nodes in each tree in the random forest, `nodesize`. The default is that trees should grow until a split would not decrease the loss, or until a split would result in a terminal node containing fewer observations than the minimum. The default node size value in `R`'s implementation, `randomForest`, is 5. Increasing this size results in greater average proximity between observations, as the probability that any two observations share a terminal node is greater.

With a single terminal node in each tree, all observations will be trivially proximate, and analogy weights will be 1. This of course would defeat the purpose of using analogy weights! If trees are grown to a full extent (the minimum terminal node size), then a lower bound on the average weight will be the $\frac{5}{\text{number of obs}}$. In practice, the average node size will be considerably larger than the minimum node size. This is because candidate splits must both decrease terminal-node variance *and* result in two terminal nodes that exceed the minimum terminal node size.

Choosing the optimal terminal node size will involve making a trade-off between including potentially irrelevant (or misleading) histories for larger values of `modesize`, and having more uncertainty in estimates for smaller values of `nodesize`. The choice should be influenced by the number of parameters in the economic model being estimated, so that on average there are a sufficient number of effective observations (ie. the average weight multiplied by the total number of observations) to provide guidance for parameter values.

```
## pdf
##   2
```

This calculation is not straightforward; using analogy weighting can both increase and decrease the precision of parameter estimates. Because analogy scores discount dissimilar observations,

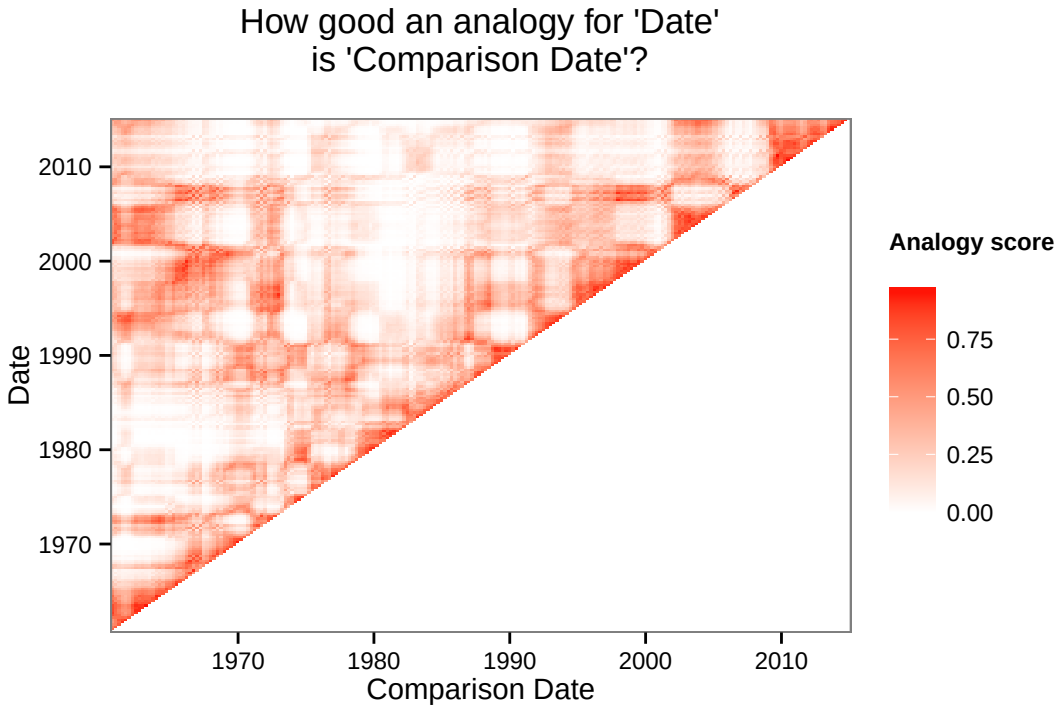
they necessarily compress the support on which the parameters are estimated (see figures XX and XX2 in the next chapter). Taken alone, this should increase uncertainty in the parameter estimates. But the entire notion of proximity weighting is that irrelevant histories are just that—and co-movements during irrelevant periods simply confound our understanding of the true relationship today. If the co-movements between dependent and independent variables during analogous histories are less variable, then the precision of parameter estimates will *increase* under analogy weighting.

An example of estimating proximity weights

The application below illustrates how one could use analogy weights in estimating the parameters of a simple economic model, and discusses the consequences of modelling choices.

The procedure of building analogy weights discussed above was carried out on quarterly US data, specifically, the unemployment rate, the rate on short-term US Treasuries, and the quarter-on-quarter percentage change of the major cities consumer price index, taking T to be the first quarter of 2015. Random forests were trained on each of these variables, taking the first and second lags of itself and the other variables to be predictors, and the out-of-bag proximity matrices were saved. The terminal node size used was 40, about a fifth of the sample; the average proximity between points should be roughly a fifth.

The plot below shows the resulting proximity matrix. Along the vertical axis is the date for which we would like historical analogies. Along the horizontal axis is the period being evaluated for similarity. The shading indicates the degree of similarity.

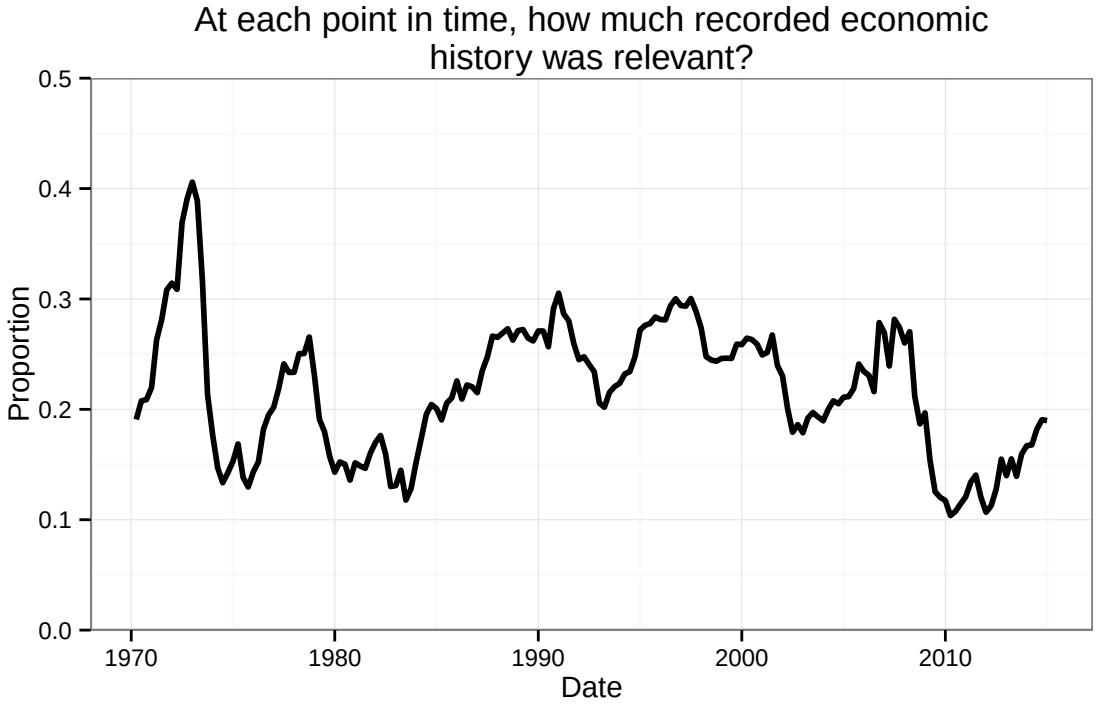


Interestingly, there appear to be three or four distinct states; the first being the post-GFC period in the top right. The second is a period of relative tranquility, roughly coinciding with the period between the early 1990s recession and the GFC, and the period before the stagflation of the 1970s. The third period, first making itself apparent in the late 1960s and continuing through until the early 1990s, is a less defined period. The emergence of fairly discrete states is interesting, given that there is no such restriction being placed on their discovery.

Another important feature of the proximity matrix is its ability to inform the user when the current state of the world is without precedent in terms of the variables that help to generate predictions. When the state of the world differs substantially to the sample, then perhaps the implications of models estimated from (a very different) history should be trusted less.

A metric to evaluate the similarity of the current state to recorded history is to divide the effective sample size (the sum of weights) by the naive sample size (the number of observations) up to a given period. This gives the effective proportion of history up to a period that would be included if you were to use the analogy weights. It is a crude measure for similarity to all

of recorded history. It is illustrated in the figure below.



Note the three periods with the largest drop in relevant histories: the oil crisis of the 1970s, the recession of the early 1980s, and the Global Financial Crisis of the 2000s. Using analogy weights during these periods would involve learning less from history, and so being more unsure of model outputs. This feature is elaborated in the portfolio risk modelling example below.

Incorporating analogy weights into empirical models

How can analogy weights be used to in estimating parameteric models? This will depend on the sort of model being estimated.

In simple models estimated using maximum likelihood, weights are typically incorporated by scaling the likelihood contribution of each data point by the weight. That is, for some density function that depends on data y and parameters θ, $f(y;\theta)$, the maximum weighted likelihood estimate using a weight vector ω_i will be

$$\text{MLE}(\theta) = \text{argmax}_\theta \sum_i \omega_i \log f(y_i; \theta)$$

For historical data points that are given little weight, this has the effect of reducing their importance in the estimation of θ. If a data point is given zero weight, it has no impact on the estimation of the parameter. It is important to adjust the standard errors of parameters estimates when using weighted maximum likelihood. They will be affected by the weights in two ways: first, the variance-covariance matrix of θ will be a weighted variance covariance matrix, and second, there will be fewer effective observations. Analytically, the (weighted) Fisher Information matrix

$$I_\omega(\theta) = -\text{E}\left[\frac{\partial^2}{\partial \theta^2} \sum_i \omega_i \log f(y_i; \theta)\right]$$

helps to estimate the weighted covariance matrix

$$\Sigma_\beta \approx \frac{1}{\sum_i \omega_i} I_\omega(\theta)^{-1}$$

The Hessian matrix in side the square brackets of the first expression tell us about the curvature of the likelihood profile at the estimated point. If the likelihood is quite flat at the estimated θ, then there should be more uncertainty about the parameter estimate. In this case, entries in the Hessian will be small, and its inverse will be large, increasing standard errors. If there is little weight placed on most historical values, then $\frac{1}{\sum_i \omega_i}$ will be large, increasing standard errors.

For more complex models estimated using Bayesian techniques, incorporating weights is slightly more involved. The below describes Hamiltonian Monte Carlo (the method used to estimate two models below) and how it can be augmented to include weighted observations.

Weighted Hamiltonian Monte Carlo in Stan

In Bayesian analysis, the aim is to conduct inference about a posterior distribution of parameters given data y and controls X $p(\theta|y, X)$. The posterior distribution is proportional

to a prior distribution of the parameters multiplied by the likelihood of the data given the parameters $p(\theta|y, X) \propto p(\theta) \times p(y|theta, X)$. The prior distribution is chosen by the researcher and incorporates information known before observing the data, such as previous study results, hard constraints on the parameter value, or an identifying belief that the parameter value is close to zero. To make inference about a parameter, the posterior needs to be integrated. For example, the mean of some parameter β would be $\hat{\beta} = \int_\beta \beta \times p(\beta|y)d\beta$. For all but a few examples, posterior distributions cannot be integrated analytically.

The solution under Markov Chain Monte Carlo is to provide draws from the posterior, which allows numerical integration to be used. That is, if an N-long "chain" of parameter esimates β_i can be drawn from the true posterior, then any given statistic about that parameter can be calculated. For example, an estimate of the parameter mean can be estimated using $\bar{\beta} = \frac{1}{N}\sum_i \beta_i$. A more detailed description including the Markov Chain theory behind the technique can be found in Greenberg (2012).

Hamiltonian Monte Carlo (as described in Betancourt and Girolami 2013 and the Stan Reference Manual 2015) provides draws from the posterior by specifying a joint probability between parameters θ and an auxiliary variable ρ, $p(\rho, \theta|y, X) = p(rho|theta, X, y) \times p(\theta|X, y)$. In most applications ρ comes from a multivariate normal distribution with covariance matrix Σ, $\rho \sim \mathcal{MVN}(0, \Sigma)$ and so is independent of θ, x, y. Borrowing from Hamiltonian Mechanics, the joint distribution defines a Hamiltonian:

$$H(\rho, \theta) = -\log p(\rho, \theta) = -\log p(\rho) - \log p(\theta|y, X) = T + V$$

Where T is the "kinetic energy" and V is the "potential energy". Hamilton's equations describe how a Hamiltonian evolves over time:

$$\frac{\partial \theta}{\partial t} = +\frac{\partial T}{\partial \rho}$$

$$\frac{\partial \rho}{\partial t} = -\frac{\partial V}{\partial \theta}$$

(the second equation holds as long as ρ is independent of the data and parameters). This system defines a two-state differential equation, which can be integrated using the leapfrog integrator, another tool borrowed from physics. Taking the initial position to be the current draw of θ, a draw of ρ is made from $\mathcal{MVN}(0, \Sigma)$. Initialised with these values, the leapfrog integrator proceeds for L steps, each of length ϵ, according to the following:

$$\rho \leftarrow \rho - \frac{\epsilon}{2}\frac{\partial V}{\partial \theta}$$

$$\theta \leftarrow \theta + \epsilon \Sigma \rho$$

$$\rho \leftarrow \rho - \frac{\epsilon}{2}\frac{\partial V}{\partial \theta}$$

This integration has the effect of "steering" θ towards parts of its distribution with higher density (unlike gradient descent, θ does not converge to the summit of the distribution, but continues over it with a given momentum). Consequently, θ spends a proportionately large amount of time towards the more dense region of its distribution, and less time at the tails. At the end of L steps, there will be some numerical error. This is because while the surface of the density of θ is smooth, movements along it in each step are straight (the error increases with step size). A Metropolis step is therefore used to decide whether to keep the draw, where the probability of keeping a draw that started at (ρ_0, θ_0) and arrived at (ρ_L, θ_L) after L steps is:

$$\min(1, \exp(H(\rho_0, \theta_0) - H(\rho_L, \theta_L)))$$

A vector of weights ω can be included in Hamiltonian Monte Carlo by weighting each observation's impact on the potential energy term $V_\omega = -\omega \log p(\theta|y, X)$. Because the prior does not rely on the data, the involves weighting each observation's contribution to the (log) likelihood. In Stan, a software package used to implement HMC, implementing these weights simply involves the following call in the model block of the program:

```
// likelihood (priors are defined above)
```

```
for(i in 1:number_of_observations){
  increment_log_prob(weight[i]* a likelihood expression of data and parameters)
}
```

Estimating models by HMC provides several advantages over other Bayesian techniques, such as Gibbs sampling or Metropolis Hastings (MH)—techniques described well in Tanner and Wong (1987), Gelfand and Smith (1990), and Chib and Greenberg (1995). The biggest is that because in a single iteration, θ may plausibly transition from any part of the the posterior to any other, draws are far more likely to be independent (see Betancourt and Girolami 2013). This is not the case in Gibbs or Metropolis Hastings. Because draws are more likely to be independent, far fewer iterations are needed to draw inference about θ. The other large advantage is that HMC is able to explore enormous parameter spaces—in the order of tens to hundreds of thousands of dimensions—without much trouble. This makes it feasible (if a little slow) to estimate time-varying parameter models by drawing directly from the posterior of the varying state space, rather than using work-arounds like the Kalman Filter. This saves time in model building, and provides more flexibility in specifying richer models (for instance, allowing the state to evolve with non-normal innovations).

Bayesian models can be estimated efficiently using the Stan software package (Stan development team, 2015), which uses an implementation of HMC known as the No U-Turn Sampler (NUTS) of Hoffman and Gelman (2014). This algorithm automatically chooses the three control parameters, L, ϵ and Σ, based on a series of adaptive iterations during the warm-up phase. Stan abstracts from the actual implementaion of HMC, allowing the user to instead specify the probability model being estimated. Users can call programs written in stan from within R, Python, Matlab, Julia or Stata.

2a: An application to estimating core inflation

Most central banks have a mandate to fight inflation, either explicitly, as with a numerical inflation target, or implicitly, as with the Fed's "just do it" approach (Mishkin, 2002). The inflation that is typically targetted is a notion of "core inflation", the inflation that would

remain after self-correcting blips have been ignored. This is a latent quantity, and must be estimated. Various methods of estimating core inflation exist—several are discussed below.

Core inflation

Core inflation is a widely-used concept that lacks a formal definition or common tool for estimation, though two conceptual approaches are common.

The first commonly-used approach is to model core inflation as being some latent factor that drives observed price series. For instance, Kapetanias (2002) defines core inflation as the first factor from a singular value decomposition of a large number of component price series. This is similar to the approach of Stock and Watson (2008), who model core inflation using a dynamic factor model that allows for structural breaks. Simpler variants of this modelling philosophy work by filtering or smoothing price series, or by simply taking averages across price series that have had surprising components removed ad-hoc (see Quah and Vahey, 1995).

A second, more principled approach is to model core inflation as being the permanent component of inflation, and therefore the part that has no effect on medium or long-run economic activity. Quah and Vahey (1995), Ribba (2003) and Cogley and Sargent (2001, 2005) take this approach. Cogley and Sargent's (2001, 2005) method is to take core inflation to be the median of posterior predictive distributions for inflation into the distant future, using a Bayesian VAR with time-varying parameters. Quah and Vahey (1995) define core inflation to be the rate of inflation that has no impact on middle to long-run growth (estimated using a restricted classical VAR).

There are problems with both approaches. The dynamic factor model approach does not typically discriminate between periods in which a signal variable contains a lot of information about the unobserved factor and when it contains less. It is also sensitive to the choice of variables used as inputs to the factors. By dropping or censoring signal series, ad-hoc approaches may not have this problem, however are both costly and atheoretic. In the second approach, an economic model must be built that provides the best possible (long-run) forecasts, given information available at the time the forecast is being made. Thus the problem

in estimating core inflation using this approach is one of finding a good forecasting model, and parameterising it with the most relevant parameters at the time of making forecasts.

In the demonstration below, I use the second approach to estimating core inflation. The demonstration makes use of three models: the first is a vanilla VAR with static parameters. The second is a time-varying parameter VAR based on the Cogley and Sargent's (2001) model. The third model is a vanilla VAR augmented to make use of analogy weights.

Three approaches formally

Vanilla Bayesian VAR

The model describes the co-movement between inflation, unemployment and the yield on short-term Treasuries, using a VAR(2) specification. Taking Y_t to be vector containing all variables, the first model is

$$\mathbf{Y_t} = \mu + \beta_\mathbf{1}\mathbf{Y_{t-1}} + \beta_\mathbf{2}\mathbf{Y_{t-2}} + \epsilon_\mathbf{t}$$

where μ and the βs are coefficients and the multivariate residual $\epsilon \sim \mathcal{MVN}(0, \Sigma)$. The parameters are estimated in a Bayesian framework, and have so-called "shrinkage" priors applied. These are all equal to 0 except the AR(1) terms, which have a prior of 1. The residual matrix Σ is decomposed into a correlation matrix Ω and a scale vector τ such that

$$\Sigma = \text{diag}(\tau)\Omega\text{diag}(\tau)$$

Under this setup, we give independent priors to the correlation and scale: the scale vectors have a prior distribution of $\tau \sim \text{Cauchy}_+(0, 2.5)$, and the correlation comes from the the LKJ distribution $\Omega \sim \text{lkj}(1.5)$. The LKJ distribution is a quite new distribution for correlation matrices that addresses some of the shortcomings of the popular Inverse-Wishart prior. The single parameter of the distribution dictates how much cross-correlation the distribution will produce; a value of 1 results in a uniform prior, while as this degrees of freedom parameter

goes to infinity, the distribution collapses on the unit-diagonal correlation matrix.

I experimented with two approaches to modelling the covariance between the coefficients μ, β_1 and β_2. Estimating a full covariance matrix would be too burdonsome given the amount of data available (it would involve estimating 210 parameters in the covariance matrix alone). One approach is to use a Littermann prior, which restricts Ω to be diagonal and specifies priors on τ. Another approach is to estimate only the covariances within each "block"; that is, allow μ to have an estimated covariance, and likewise with the βs. This has the advantage that it only requires estimating 78 covariance parameters, and was the approach taken for this model.

The model is estimated using HMC as described above, with no weights.

While being fitted, the model calculates 1-year-ahead posterior predictions; these make use of both the uncertainty in the regression error and uncertainty in parameter estimates. These predictions form the basis for the model comparisons below.

Weighted VAR

The second model fitted takes exactly the same form as the VAR described above, with one difference: it is fitted using analogies weights. These weights are calculated as in Section XX above, using inflation, unemployment and the interest rate as the variables used to estimate analogies.

Time-varying parameter VAR

The third model considered is a Time-varying-parameter VAR, similar to the one described in Cogley and Sargent (2001). The model is

$$\mathbf{Y_t} = \mu_\mathbf{t} + \beta_{\mathbf{1},\mathbf{t}}\mathbf{Y_{t-1}} + \beta_{\mathbf{2},\mathbf{t}}\mathbf{Y_{t-2}} + \epsilon_\mathbf{t}$$

with the coefficients themselves evolving over time. Take $\theta = [\mu, \text{vec}\beta_1, \text{vec}\beta_2]'$, where the vec operation stacks the a matrix in column-major order, then:

$$\theta_t = \theta_{t-1} + \eta_t$$

and $\eta \sim \mathcal{MVN}(0, \Sigma_\eta)$. For identifiability, suitable structure must be put on Σ_η; in the work illustrated below, I assume that it is diagonal.

Data used

The data used come from the vintage data releases contained in the ALFRED database. The data series are: The annualised CPI growth rate for major cities; the national non-farm unemployment rate; and the annualised yield on 90 day Treasuries.

Forecasting performance of the three approaches

2b: An application to portfolio risk management

Portfolio risk managers are tasked with holding sufficient cash to insure that asset purchases and withdrawals from the portfolio can be paid for even when asset prices fall. This requires generating estimates of Value At Risk (VaR), a measure defined as the minimum loss expected in some event that should occur with a given frequency over a period. For example, if a portfolio has \$5M VaR over 1 day at 95%, this would mean that not more than one in twenty days should involve losses greater than \$5M.

Due to volitility clustering (Mandlebrot 1963), the VaR of a portfolio will change over time. More volatile periods will imply a greater chance of loss, and so VaR will increase. A good volatility model can help managers hold liquidity at times that are most likely to be most volatile. Generalized Auto-Regressive Conditional Heteroskedasticity (GARCH)-style models (due to Bollerslev 1986) are a common choice for this task.

For a stationary univariate time series y_t, a GARCH(p,q) model (for simplicity with no lags of y) has the following specification:

$$y_t = \mu + h_t \eta_t$$

Where μ is the unconditional mean value of y, η is i.i.d. around 0, and h_t is the conditional standard deviation of residuals. h_t evolves according to:

$$h_t^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i (y_{t-i} - \mu)^2 + \sum_{j=1}^{q} \beta_j h_{t-j}^2$$

For a given starting value h_0, estimating a GARCH model requires estimating only μ, a $p+1$-dimensional vector α, and a q-dimensional vector β. This is most commonly done using maximum likelihood, and can be estimated using Bayesian techniques also.

While univariate GARCH models offer a simple method of modelling conditional volatility, often a risk manager may be concerned about the correlation between assets' returns. The Constant Conditional Correlation (multivariate) GARCH model of Bollerslev (1990) provides a very simple method of modelling this correlation. A vector of stationary variables $\mathbf{y}_t$ with unconditional mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{H}_t$ behaves according to

$$\mathbf{y}_t = \boldsymbol{\mu} + \mathbf{H}^{1/2} \boldsymbol{\eta}_t$$

Where $\boldsymbol{\eta}$ is iid with a unit diagonal covariance matrix. The covariance matrix is modelled as being the diagonal quadratic form of a scale vector that varies over time $\boldsymbol{\tau}_t$ and a correlation matrix that is fixed, $\boldsymbol{\Omega}$.

$$H_t = \text{diag}(\boldsymbol{\tau}_t) \boldsymbol{\Omega} \text{diag}(\boldsymbol{\tau}_t)$$

And the scale parameters themselves are modelled as coming from a GARCH process.

$$\boldsymbol{\tau}_t \odot \boldsymbol{\tau}_t = \boldsymbol{\omega} + \sum_{i=1}^{p} \mathbf{A_i} (\boldsymbol{y}_{t-i} - \boldsymbol{\mu}) \odot (\boldsymbol{y}_{t-i} - \boldsymbol{\mu}) + \sum_{j=1}^{q} \mathbf{B}_j \boldsymbol{\tau}_{t-1-j} \odot \boldsymbol{\tau}_{t-1-j}$$

Here, $\odot$ is elementwise multiplication. $\boldsymbol{\omega}$ is a vector of intercept terms, while $\mathbf{A}$ and $\mathbf{B}$ are

diagonal matrices of parameters.

CCC-GARCH models offer an improvement beyond simple GARCH models of volatility in that they allow correlation between series. Yet for a range of tasks, the fixed correlation structure is too restrictive. As outlined in Engel (1999), a wide range of tasks in financial manegememt require updated estimates of the correlation between variables. This necessity has led to a large number of GARCH-style models that have time-varying correlation matrices, including popular methods like VC-GARCH of Tse and Tsui (2002) and DCC-GARCH of Engle (1999, 2002), in which correlations react to recent comovements.

The analogy-weighted CCC-GARCH model described below improves on Bollerslev's (1990) model by making two changes. First, it allows for analogy weighting. The practical impact of allowing analogy weights is to give time-variation to the correlation matrix, so that the model will have similar correlations between asset returns across similar states of history.

The second big improvement is an LKJ prior for the correlation matrix with a small degrees of freedom parameter. This has the effect of "switching off" the model when the world alters states to a new (unprecedented) state, imposing large correlations between returns, increasing VaR. It does this because an unprecendented state has little in common with historical observations, and so the model draws more inference from the prior. Because the LKJ prior is a uniform correlation matrix when the degrees of freedom parameter is 1, relying more on the prior implies assuming more correlation between observations.

A modern CCC-GARCH model and the analogy augmentation

The functional form of of the augmented CCC-GARCH model is identical to the one as described above—all that differs is the estimation technique. To provide Bayesian inference about the model, prior distributions are given to the parameters. Elements of $\boldsymbol{\mu}$, diag($\mathbf{A}$) and diag($\mathbf{B}$) are all given univariate normal priors. For stability, it is necessary for elements of diag($\mathbf{A}$) to be greater than 1, and diag($\mathbf{B}$) $\leq \mathbf{I} -$ diag($\mathbf{A}$). The initial values of the scale parameter, τ_0, are given half Cauchy priors, which allow for large initial variances. Finally, the correlation matrix Ω is given an LKJ prior with a degrees of freedom parameter close

to 1; this has the effect of increasing modelled correlations during unprecedented phases. Weights are incorporated as described above—by weighting each observation's contribution to the log posterior.

To demonstrate the performance of the two techniques, an equal-weighted portfolio is constructed from common stocks in Apple (AAPL), IMB (IBM), Google (GOOGL), Ford (FORD), Bank of America (BAC), and Johnson and Johnson (JNJ). Two versions of a CCC-GARCH(1,1) model are estimated on weekly log returns in successive 1-week increments: an unweighted version, and one using analogy weights. Both models are then tasked with generating estimates of 1-week ahead 95% VaR for the second half of 2008 using true out-of-sample prediction. That is, both models are estimated using data from August 22, 2004 through June 29, 2008, and produce VaR estimates for the week ended July 6 2008. For the next iteration, models are estimated from the same start-date up to July 6, and make another 1-week-ahead forecast. This is continued until the end of 2008; the test period covers the most turbulent period of the Global Financial Crisis of 2007-8.

Week-ahead VaR estimates are generated by generating 1-week ahead posterior predictive distributions for all stocks in the portfolio. The posterior predictive distribution is similar to a classical forecast, but allows for uncertainty in the model parameters. In each HMC draw, the projections for each stock are aggregated to construct a portfolio return; the VaR measure is then the bottom 5th percentile of the resulting distribution. For a correctly specified model, losses should not exceed VaR more than one in twenty weeks.

The analogy weights are constructed using the technique described above, using random forests to predict the weekly squared returns of each of the stocks in question, using their own (squared) first and second lags, and the (squared) first and second lags of the other stocks.

The unconditional mean prior is $\boldsymbol{\mu} \sim \mathcal{N}(0, 0.01)$, indicating that we expect the unconditional weekly return to be close to 0 over long periods. The volatility scale prior is $\boldsymbol{\tau} \sim \text{Cauchy}_+(0, 2.5)$, while the prior on elements of $\boldsymbol{\mu}$, $\mathbf{A}$, and $\mathbf{B}$ are all $\mathcal{N}(0, 1)$.

Predictive performance

Performance of vanilla MGARCH and analogy-weighted MGARCH during crisis

5: Conclustion