The debate about business confidence as a possible leading indicator of economic growth reminded me of a question that has long been on my back-burner – what data make the best leading indicators (say, a quarter of a year ahead) of economic growth? There are a number of indicators that get a lot of attention in the media and from economic pundits of course, but I wanted to take a data-driven approach and see what actual evidence there is for the various candidate variables.

To cut to the chase, this graphic summarises my results:

The previous quarter’s GDP growth is, unsurprisingly, a strong indicator of the coming quarter’s. Food prices are strongly negatively related to economic growth (in New Zealand anyway – with the strong agriculture sector here food prices may be serving as a proxy for something else, but exactly what I’m not sure although I’ve got some ideas). Other indicators with fair evidence of value include visitor arrivals, car registrations, business confidence (the OECD measure, which David Hood points out is [apparently derived from the NZIER’s survey, not the ANZ’s](https://twitter.com/Thoughtfulnz/status/1027339431638749185)), and electronic card transactions.

One surprise for me is the complete absence of predictive power of merchandise goods exports for growth in GDP. Any suggestions why welcomed in the comments!

These confidence intervals come from what I believe to be a pretty robust frequentist modelling approach that combines multiple imputation, bootstrap and elastic net regularization. I have in the back of my head the idea of a follow-up project that uses Bayesian methods to provide actual forecasts of the coming GDP, but I may not get around to it (or if I do, it will probably be for Victorian or Australian data seeing as I’ll be living there from October and more motivated for forecasts in that space).

**Data**

To address the question, I started with the [Stats NZ data release calendar](https://www.stats.govt.nz/release-calendar/) to identify official statistics that are available on a monthly basis and published within a month of the reference period. I counted seven such collections:

* Electronic card transactions
* Food price index
* Transport vehicle registrations
* International travel and migration
* Overseas merchandise trade
* Building consents issued
* Livestock slaughtering statistics

To these seven, I also added the:

* Reserve Bank’s Trade Weighted Index of the New Zealand dollar compared to a basket of currencies
* OECD business confidence series (I used this rather than the ANZ or NZIER original series as the OECD version is available freely for reproducibility purposes)

Some of these datasets are available programmatically, and some required manual steps to download from Infoshare. I set up for the project, as are all the scripts that import and combine the data.

Here’s the original time series, plus seasonally adjusted versions of all of them except for the trade weighted index (currencies are so random I thought it made little sense to do a seasonally adjusted version of this; possible subject of future investigation):

I did my own seasonal adjustment because even when seasonally adjusted series are available in some of the Stats NZ collections, typically it goes back only to the 1990s or 2000s. I wanted data at least all the way back to 1987 when the quarterly GDP series starts; time series are difficult enough to analyse as it is without restricting ourselves to a decade or so of data (this is also why I went back to original data series rather than using the Treasury “leading economic indicator” Excel collections, as they cover only a relatively recent period).

Apart from business confidence (which by its nature is restricted to within a constant range), the original data are all clearly non-stationary, so I converted them to growth rates, which gives us the following chart summarising my candidate leading economic indicators:

Both seasonal adjustment and stationarity are important with time series so we can focus on the underlying relationships and avoid spurious correlations from the structural (growth and episodicity) elements of the series. An alternative approach would be to model the seasonality and growth directly in a full blown Bayesian model, but even in that case I’d want to look at de-seasoned, stationary versions for the exploratory stage.

Now that we have those de-seasoned, stationary versions of the data, we can look at them and see which ones are correlated one-on-one with economic growth for the quarter. Remember, the idea here is to use data that are available fairly promptly as leading indicators of economic growth in the same quarter, because economic growth is only reported quarterly and is hard to estimate, and hence is reported at a material lag. Here are the bivariate correlations:

I’ve sequenced the variables in order of strength of relationship with GDP growth in the quarter. We can see that lagged GDP growth, electronic card transactions growth, and OECD business confidence have the three strongest positive correlations with GDP growth. Food price growth has a strong negative correlation (ie high growth in food prices is a predictor of low growth in GDP).

|  |
| --- |
| # full set of data, in a pretty wide format: |
|  | ind\_data <- gdp\_q %>% |
|  | full\_join(bc\_q, by = c("yr", "qtr")) %>% |
|  | full\_join(ect\_q, by = c("yr", "qtr")) %>% |
|  | full\_join(bci\_q, by = c("yr", "qtr")) %>% |
|  | full\_join(fpi\_q, by = c("yr", "qtr")) %>% |
|  | full\_join(iva\_q, by = c("yr", "qtr")) %>% |
|  | full\_join(cars\_q, by = c("yr", "qtr")) %>% |
|  | full\_join(com\_vcl\_q, by = c("yr", "qtr")) %>% |
|  | full\_join(goods\_q, by = c("yr", "qtr")) %>% |
|  | full\_join(fx\_q, by = c("yr", "qtr")) %>% |
|  | full\_join(lst\_q, by = c("yr", "qtr")) %>% |
|  | arrange(yr, qtr) %>% |
|  | mutate(yr\_num = yr + qtr / 4 - 0.125) |
|  |  |
|  | # tidy format, and just the stationary versions (ie growth etc): |
|  | ind\_data\_tidy <- ind\_data %>% |
|  | # Drop some non-seasonally-adjusted variables that we won't be using: |
|  | select(-gdp, -bc, -bci, -ect, -iva) %>% |
|  | gather(variable, value, -yr, -qtr, -yr\_num) %>% |
|  | filter(!is.na(value)) %>% |
|  | mutate(type = ifelse(grepl("\_sa", variable), "Seasonally adjusted", "Original"), |
|  | lagged = ifelse(grepl("\_lag", variable), "Lagged", "Non-lagged"), |
|  | diffed = ifelse(grepl("\_growth", variable), "Growth", "Original")) %>% |
|  | filter(diffed == "Growth" | variable %in% c("bc\_sa")) %>% |
|  | filter(lagged != "Lagged" | variable == "gdp\_growth\_lag") %>% |
|  | select(-yr, -qtr) %>% |
|  | mutate(variable = fct\_relevel(variable, c("gdp\_growth", "gdp\_growth\_lag", "ect\_growth", "bc\_sa", |
|  | "cars\_growth", "com\_vcl\_growth", "iva\_growth", "bci\_growth", |
|  | "twi\_growth", "lst\_growth", "goods\_growth"))) |
|  |  |
|  | # wide again, but with only the variables we need. Useful for modelling. |
|  | ind\_data\_wide <- ind\_data\_tidy %>% |
|  | select(-lagged, -type, -diffed) %>% |
|  | spread(variable, value) %>% |
|  | mutate(yr\_num2 = yr\_num) %>% |
|  | select(-yr\_num) %>% |
|  | rename(yr\_num = yr\_num2) |
|  |  |
|  | # wide version with meaningful names: |
|  | ind\_data\_wide\_names <- ind\_data\_wide %>% |
|  | rename(`GDP growth one quarter ago` = gdp\_growth\_lag, |
|  | `Business confidence` = bc\_sa, |
|  | `Building consents` = bci\_growth, |
|  | `Electronic card transactions` = ect\_growth, |
|  | `Food price index` = fpi\_growth, |
|  | `GDP growth` = gdp\_growth, |
|  | `Visitor arrivals` = iva\_growth, |
|  | `Cars and stationwagons registered` = cars\_growth, |
|  | `Commercial vehicles registered` = com\_vcl\_growth, |
|  | `Exports of goods` = goods\_growth, |
|  | `Trade-weighted currency index` = twi\_growth, |
|  | `Number livestock slaughtered` = lst\_growth) |
|  |  |
|  | names(ind\_data\_wide\_names) <- str\_wrap(names(ind\_data\_wide\_names), 15) |
|  |  |
|  | # Save all four versions: |
|  | save(ind\_data, ind\_data\_wide, ind\_data\_tidy, ind\_data\_wide\_names, file = "data/ind\_data.rda") |

The lower diagonal in that pairs plot shows connected scatter plots with a single-predictor linear model. This turned out to be pleasantly easy to do with GGally::ggpairs and a simple user-defined function.

**Minimalist regression (poor)**

My starting point was a fairly naive out-of-the-box linear regression, just chucking all the data into lm().

load("data/ind\_data.rda")

summary(ind\_data\_wide) # 109 cases where even gdp\_growth is missing. No point in using them

ind\_data\_wide2 <- subset(ind\_data\_wide, !is.na(gdp\_growth\_lag))

# naive approach \*will\* fit a plausible model, but the NA is down to the lowest common denominator, when data available on all variables

mod <- lm(gdp\_growth ~ ., data = ind\_data\_wide2)

stargazer(mod, type ="html")

This gives me this result:

|  |  |
| --- | --- |
|  | |
|  | *Dependent variable:* |
|  |  |
|  | gdp\_growth |
|  | |
| gdp\_growth\_lag | 0.009 |
|  | (0.121) |
|  |  |
| ect\_growth | 0.335\*\*\* |
|  | (0.114) |
|  |  |
| bc\_sa | 0.0001 |
|  | (0.001) |
|  |  |
| cars\_growth | 0.022 |
|  | (0.016) |
|  |  |
| com\_vcl\_growth | 0.003 |
|  | (0.009) |
|  |  |
| iva\_growth | 0.046\*\* |
|  | (0.020) |
|  |  |
| bci\_growth | 0.001 |
|  | (0.008) |
|  |  |
| twi\_growth | -0.004 |
|  | (0.020) |
|  |  |
| lst\_growth | -0.002 |
|  | (0.008) |
|  |  |
| goods\_growth | 0.005 |
|  | (0.012) |
|  |  |
| fpi\_growth | -0.236\*\*\* |
|  | (0.080) |
|  |  |
| yr\_num | 0.0001 |
|  | (0.0002) |
|  |  |
| Constant | -0.177 |
|  | (0.354) |
|  |  |
|  | |
| Observations | 61 |
| R2 | 0.566 |
| Adjusted R2 | 0.457 |
| Residual Std. Error | 0.004 (df = 48) |
| F Statistic | 5.215\*\*\* (df = 12; 48) |
|  | |
| *Note:* | \*p<0.1; \*\*p<0.05; \*\*\*p<0.01 |

There’s two big problems with this approach:

* By default, the model is fit to the minimal set of the data that is complete for all variables. Electronic card transactions is the shortest series, going back only to 2003 when it is treated as a growth rate. So all the information in the other variables before that time has been thrown out.
* A big amount of collinearity in the data is inevitable with this sort of collection. While it doesn’t bias the estimated coefficients, it can make a big increase to the variance, and basically adds a lot of noise to our estimated effect sizes. I think this is why we have the implausible result in this naive model of lagged GDP growth not being useful as an explanatory variable.

… and one small problem, which is that other than including the lagged value of GDP growth as an explanatory variable, I’m ignoring the time series nature of the data. More on that later.

**Multiple imputation (better)**

We could address the problem of the short electronic card transactions series either by eliminating that series from the data altogether (which would be a shame as it’s obviously useful), or by imputing it’s previous values. Conceptually this is a little dubious – what exactly are we saying growth in electronic card transactions in 1988 would mean? – but statistically it’s fine. The imputed values will capture the relationships between the various variables from 2003 onwards and avoid creating new artefacts, so in effect we will find the relationship between electronic cards and the others for the years we have the data, while still using the full 1987+ data for all other variables.

The appropriate way to do this is to create multiple complete data sets with different imputed values, fit our model to each, and pool the results. The mice package makes this spectacularly easy:

ind\_mi <- mice(ind\_data\_wide2, print = FALSE)

mod\_mi <- with(ind\_mi, lm(gdp\_growth ~ yr\_num + gdp\_growth\_lag + bc\_sa + bci\_growth +

ect\_growth + fpi\_growth + iva\_growth + goods\_growth +

cars\_growth + com\_vcl\_growth + twi\_growth + lst\_growth))

summary(pool(mod\_mi)) %>% kable(digits = 4)

|  | **estimate** | **std.error** | **statistic** | **df** | **p.value** |
| --- | --- | --- | --- | --- | --- |
| (Intercept) | -0.2889 | 0.1712 | -1.6878 | 54.8951 | 0.0944 |
| yr\_num | 0.0001 | 0.0001 | 1.5128 | 59.0142 | 0.1333 |
| gdp\_growth\_lag | 0.2182 | 0.0846 | 2.5785 | 79.8727 | 0.0113 |
| bc\_sa | 0.0005 | 0.0004 | 1.3342 | 100.3535 | 0.1850 |
| bci\_growth | 0.0061 | 0.0069 | 0.8820 | 106.1382 | 0.3798 |
| ect\_growth | 0.2059 | 0.1308 | 1.5741 | 15.6965 | 0.1184 |
| fpi\_growth | -0.1900 | 0.0566 | -3.3599 | 105.0452 | 0.0011 |
| iva\_growth | 0.0257 | 0.0193 | 1.3335 | 82.9585 | 0.1852 |
| goods\_growth | 0.0008 | 0.0125 | 0.0628 | 97.7269 | 0.9500 |
| cars\_growth | 0.0172 | 0.0071 | 2.4403 | 104.3330 | 0.0163 |
| com\_vcl\_growth | 0.0050 | 0.0077 | 0.6501 | 102.0428 | 0.5171 |
| twi\_growth | 0.0150 | 0.0192 | 0.7807 | 103.6387 | 0.4367 |
| lst\_growth | 0.0024 | 0.0061 | 0.3999 | 104.7962 | 0.6900 |

Compared to the naive model on the cut-down dataset, we now have a much more plausible strong relationship with lagged GDP growth (which didn’t show up at all as significant in the first model). Food prices and car/stationwagon registrations the other two variables with “significant” evidence at this stage. The electronic card transactions (ect\_growth) effect is much smaller now, an unsurprising result from the “blurring” of the data introduced by the imputation approach and the fact that the other variables have long series of actual values to assert their relationships with the response of GDP growth.

So, we’ve dealt with the missing data, but not yet the multi-collinearity.

**Multiple imputation, elastic net regularization, and bootstrap (best)**

My preferred approach to reducing the random noise in effect sizes from having collinear explanatory variables is elastic net regularization, which I’ve written about a number of times in this blog.

Regularization can be combined with multiple imputation if we use a bootstrap. For each bootstrap re-sample, we perform a single imputation of missing variables, using just the data in the re-sample. This adds a little to the computational complexity, but this dataset isn’t large, and the more valid inferences possible make it worthwhile. In general, as much of the data processing stage as possible should be repeated in each bootstrap re-sample, for the re-sampling process to work as intended, ie as a mimic of “what would happen if we could repeat this experiment 999 times”, the fundamental principle of frequentist inference.

Bootstrap of a time series normally needs special methods. I’m not necessarily happy with the shortcut I’ve taken here, which is to say that I can treat the data as cross-sectional other than by including the lagged GDP growth variable. However, with all the variables in their stationary state, and with the residual series not showing signs of autocorrelation (more on this later), I think it’s fair enough in a pragmatic way.

I was able to cannibalise code from my previous posts with minimal amendments.

My first step is to estimate good values for the alpha hyper-parameter in generalised linear regression. This is the value between 0 and 1 that lets you choose a combination of the *type* of penalty used to determine the amount that coefficients are shrunk towards zero, to counter the way collinearity adds random noise to their estimates. I do this by fitting models with a range of different values of alpha, while using cross-validation (built into the glmnet package) to get good values of lambda. lambda is the hyper-parameter that indicates how *much* shrinkage there is.

# I like Michael's answer at https://stats.stackexchange.com/questions/46719/multiple-imputation-and-model-selection:

# use in combination with bootstrap.

cv\_results <- data\_frame(lambda = numeric(), alpha = numeric(), mcve = numeric(), imputation = numeric())

set.seed(765)

for(imp in 1:5){

the\_data <- mice::complete(ind\_mi, imp)

X <-as.matrix(select(the\_data, -gdp\_growth))

Y <- the\_data$gdp\_growth

alphas <- seq(from = 0, to = 1, length.out = 21)

for(i in alphas){

cvfit <- cv.glmnet(as.matrix(X), Y, alpha = i)

tmp <- data\_frame(lambda = cvfit$lambda, alpha = i, mcve = cvfit$cvm, imputation = imp)

cv\_results <- rbind(cv\_results, tmp)

}

}

cv\_res\_grp <- cv\_results %>%

group\_by(lambda, alpha) %>%

summarise(mcve = mean(mcve)) %>%

ungroup() %>%

arrange(mcve)

cv\_res\_grp %>%

mutate(ordered\_alpha = as.factor(round(alpha, 3)),

ordered\_alpha = fct\_reorder(ordered\_alpha, mcve, .fun = min)) %>%

ggplot(aes(x = lambda, y = sqrt(mcve), colour = ordered\_alpha)) +

geom\_line(size = 2) +

scale\_x\_log10(label = comma) +

scale\_colour\_viridis("alpha", guide = guide\_legend(reverse = TRUE), discrete = TRUE) +

ggtitle("Cross-validation to select hyper parameters in elastic net regression") +

scale\_y\_continuous("Square root of mean cross validation error", label = comma) +

theme(legend.position = "right")

chosen\_alpha <- arrange(cv\_res\_grp, mcve)[1, ]$alpha

This gives me a graphic like the following:

We can see that we can get low values of mean error with many combinations of alpha and lambda, which is reassuring in suggesting that the choices aren’t critical. But I store the best value of alpha for use in the actual bootstrapped model fitting to follow.

Here’s the code that does the actual bootstrapped, mulitply-imputed, elastic net regularized fitting of my model. It has four steps:

* define a function elastic() that takes a resampled set of data, performs a single imputation on it, determines a good value of lambda, fits the model and returns the values of the coefficients
* use the boot::boot function to run that function on 999 different samples with replacement of the original data
* tidy up the resulting percentile confidence intervals into a neat data frame
* draw a nice graphic showing the confidence intervals

Most of the code below is actually in the third and fourth of those steps:

# Define a function that creates a single imputed set of data from a bootstrapped resample, then fits ridge regression to it

elastic <- function(data, i){

# create a single complete imputed set of data:

the\_data = mice::complete(mice(data[i, ], m = 1, print = FALSE), 1)

X <- as.matrix(dplyr::select(the\_data, -gdp\_growth))

Y <- the\_data$gdp\_growth

# we're going to scale the data so the results are in terms of standard deviations of the

# variable in question, to make coefficients more comparable. Note that this scaling takes

# place \*within\* the bootstrap as it is part of the analytical process we are simulating

# by resampling.

X <- scale(X)

# work out the best lambda for this dataset using cross validation"

lambda <- cv.glmnet(as.matrix(X), Y, alpha = chosen\_alpha)$lambda.min

# fit the model:

mod\_glmnet <- glmnet(X, Y, alpha = chosen\_alpha, lambda = lambda)

# return the results:

as.vector(coef(mod\_glmnet))

}

# Now we run the bootstrap, using the function above.

set.seed(321)

elastic\_bt <- boot(data = ind\_data\_wide2, statistic = elastic, R = 999)

# process results into a tidy data frame

boot\_coefs <- data\_frame(

variable = character(),

lower = numeric(),

upper = numeric(),

point = numeric()

)

var\_names <- c("Intercept", names(ind\_data\_wide\_names)[-2])

for(i in 1:length(var\_names)){

x <- boot.ci(elastic\_bt, type = "perc", index = i)

boot\_coefs <- rbind(boot\_coefs,

data\_frame(variable = var\_names[i],

lower = x$percent[4],

upper = x$percent[5],

point = x$t0))

}

# draw graphic

boot\_coefs %>%

filter(!variable %in% c("yr\_num", "Intercept")) %>%

# next line is in there in case we do still want the time trend in the graphic. But it's never significant,

# and it's hard to explian.

mutate(variable = ifelse(variable == "yr\_num", "Linear\ntime trend", variable)) %>%

mutate(variable = fct\_reorder(variable, lower)) %>%

ggplot(aes(y = variable)) +

geom\_vline(xintercept = 0, colour = "black") +

geom\_segment(aes(yend = variable, x = lower, xend = upper), size = 3, colour = "steelblue", alpha = 0.5) +

geom\_point(aes(x = point)) +

scale\_x\_continuous(label = percent) +

labs(x = "Estimated impact in percentage points on coming quarter's GDP growth,

of change in one standard deviation in the explanatory variable",

y = "",

caption = "95% confidence intervals based on bootstrapped elastic net regularized regression, with

electronic card transactions imputed differently for each bootstrap resample.

Analysis by http://freerangestats.info") +

ggtitle("Previous quarter's economic growth (+ve) and food prices (-ve)

most useful as leading indicators of this quarter's New Zealand economic growth",

str\_wrap("Variables considered are official statistics available from Stats NZ every month, within a month; plus the OECD business confidence measure (which is based on NZIER's Quarterly Survey of Business Opinion); and the trade weighted index for currency published by RBNZ. Data goes back to 1987.", 80))

And I’m sticking to that (same graphic as I started the post with) as my best results.

**Timeseries method**

I’ve mentioned a couple of times my nervousness about how I’ve inadequately treated the timeseries nature of these data. Here’s a quick look at the residuals of one version of my model (ie one set of imputed values). It assures me that there is no evidence of the residuals being autocorrelated, but I still think I’m cutting corners:

lambda <- cv.glmnet(as.matrix(X), Y, alpha = chosen\_alpha)$lambda.min

mod\_glmnet <- glmnet(X, Y, alpha = chosen\_alpha, lambda = lambda)

residuals <- ts(Y - predict(mod\_glmnet, newx = X), start = c(1987, 4), frequency = 4)

# Fail to reject null hypothesis of no autocorrelation (which means that it is probably ok to have used the

# above method, pretending it is cross-sectional other than by including the lagged GDP)

Box.test(residuals, lag = 4, type = "Ljung-Box")

par(family = "myfont", font.main = 1, bty = "l")

tsdisplay(residuals, main = "Time series of residuals from regression with imputation and regularization")

As an alternative approach, I used the forecast::auto.arima() approach to model an imputed set of the data explicitly as a timeseries. This gets me some results that differ slightly from the above, but perhaps are broadly compatible:

| **.rownames** | **X2.5..** | **X97.5..** |
| --- | --- | --- |
| ar1 | 0.7812 | 1.0434 |
| ma1 | -1.8363 | -1.5284 |
| ma2 | 0.6152 | 0.9077 |
| sar1 | 0.0547 | 0.5393 |
| sar2 | -0.4780 | -0.0796 |
| sma1 | -1.0269 | -0.7013 |
| ect\_growth | 0.0003 | 0.0018 |
| bc\_sa | 0.0009 | 0.0036 |
| cars\_growth | -0.0001 | 0.0017 |
| com\_vcl\_growth | -0.0007 | 0.0010 |
| iva\_growth | -0.0003 | 0.0017 |
| bci\_growth | -0.0010 | 0.0004 |
| twi\_growth | -0.0002 | 0.0013 |
| lst\_growth | -0.0001 | 0.0016 |
| goods\_growth | -0.0011 | 0.0008 |
| fpi\_growth | -0.0023 | -0.0003 |

The key difference is that business confidence (bc\_sa) and, to a lesser extent, electronic card transactions (ect\_growth) are more strongly significant in this version. Food price (fpi\_growth) is still strongly negative, and other confidence intervals all encompass zero. Note that lagged GDP growth doesn’t appear in this model explicitly – it comes in automatically as the ar1 (auto-regression at lag 1) term of the algorithmically-chosen ARIMA model.

The results are similar enough that I decide I’m ok with the approach I’ve taken, so I don’t invest in what would be quite an effort (but eminently possible with time) to introduce multiple imputation and regularization into the time series model.

Here’s the code for that bit of time series modelling:

set.seed(234)

ind\_imp <- mice::complete(mice(ind\_data\_wide2, m = 1, print = FALSE), 1)

Y <- ts(ind\_imp$gdp\_growth, start = c(1987, 4), frequency = 4)

X <- scale(select(ind\_imp, -yr\_num, -gdp\_growth, -gdp\_growth\_lag))

mod <- auto.arima(Y, xreg = X)

confint(mod) %>% tidy %>% kable(digits = 4)