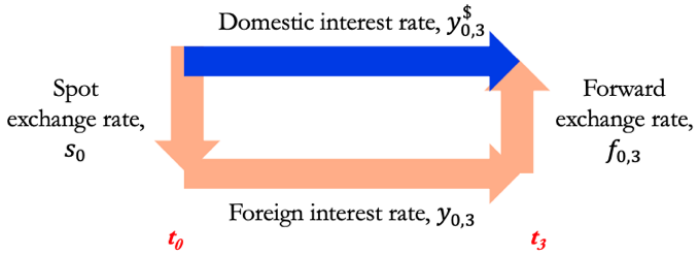
Predicting Foreign Exchange Arbitrage

Stefan Huber & Amy Wang

1 Introduction and Related Work

The Covered Interest Parity condition ("CIP") should dictate prices on the trillion-dollar foreign exchange market. As illustrated in Figure 1, since the US and foreign interest rates, spot exchange rate, and forward exchange rate are all known at the same time, it must be that the payoff from earning the dollar interest directly – doing the blue trade – is equal to the payoff from first converting the US dollar ("USD") into foreign currency at the spot exchange rate, then earning the foreign interest, and finally converting the proceeds back to USD at the known forward exchange rate – doing the orange trade.

Figure 1: **Covered (three-month) interest-rate parity condition**



CIP requires that the payoff from doing the blue trade equal to the payoff from doing the orange trade. The first subscript indicates the time when a rate is known. The second subscript indicates the duration of the rate.

Yet the CIP fails. When CIP fails, the cross-currency basis ("basis") appears. A non-zero basis, as defined by Equation 1, is an arbitrage opportunity, and is thus of great interest to finance academics and practitioners alike.

$$x_{t,0,b} = y^{\$}_{t,0,b} - y_{t,0,b} - \frac{1}{b}(s_t - f_{t,0,b}). \tag{1}$$

Our project seeks to predict the bases in the Post-2007-2009 Financial Crisis period of the Australian dollar ("AUD") and of the Japanese yen ("JPY"), both relative to USD. We purposely choose to predict both the AUD basis and the JPY basis because they tend to have opposite signs and may exhibit interesting cross-sectional relationships. Employing Regression Trees, Random Forests, and three versions of Regularized Regressions, we predict the bases with reasonable success: our best test MSEs of about 70 for AUD and 200-270 for JPY compare favorably to the variance of the test-period bases, which are 11 and 24 for AUD and JPY, respectively.

Systematic violations of CIP are first documented by Du et al. (2018). This and other works, such as Hébert and Wang (2018) and Boyarchenko et al. (2018), postulate that cross-currency bases are reflections of constraints faced by financial intermediaries. Yet there is no known effort on predicting the basis and thereby establishing quantitative links between bases and other observables. Machine learning ("ML") techniques are appropriate to fill this void. More generally, ML is starting to be used in finance research: Kozak et al. (2019) marries the theory on stochastic discount factor ("SDF") with regularized regressions on stock returns to identify factors in the SDF; Gu et al. (2018) compares the performance on stock return predictions from various ML methods, including generalized linear models, dimension reduction, boosted regression trees, random forests, and neural networks. Our work broadens the application of ML to finance, and provides empirical evidence that can illuminate theoretical constructs of asset price movements on the foreign exchange market.

2 Dataset and Features

We construct using Equation 1 the outcome variables: the AUD basis and the JPY basis. Interest rates are taken to be the fixed rate in Overnight Index Swaps against central bank policy rates. We further collect 32 data series for each of the three relevant currencies: AUD, JPY, and USD. Our data capture activities in the financial markets, conditions of the economy, the state of international trade, and the stance of economic policies (see Figure 2). With the exception of the Economic Policy Uncertainty ("EPU") Index, which is compiled by three leading economists and available at `www.policyuncertainty.com`, all data are obtained from Bloomberg.

We first pre-process the data by computing and using, for most of the features, the percentage change between two observations. This is done to both normalize and extract more meaningful information. We next augment the data in two ways. First, while all Financial Markets data are reported daily, other data are available only monthly or quarterly; for low frequency series, we impute daily observations based on the last available entry. Second, we interact and pairwise interact all features; these additional features are used in the polynomial specification of regularized regressions. With the cleaned data set, we construct two distinct samples that emphasize different aspects of the data. The *Complete* sample retains the longest possible time horizon by including series that are available between 2004 and 2017. The *Post-Crisis*

Figure 2: **List of features in each of AUD, JPY, USD**

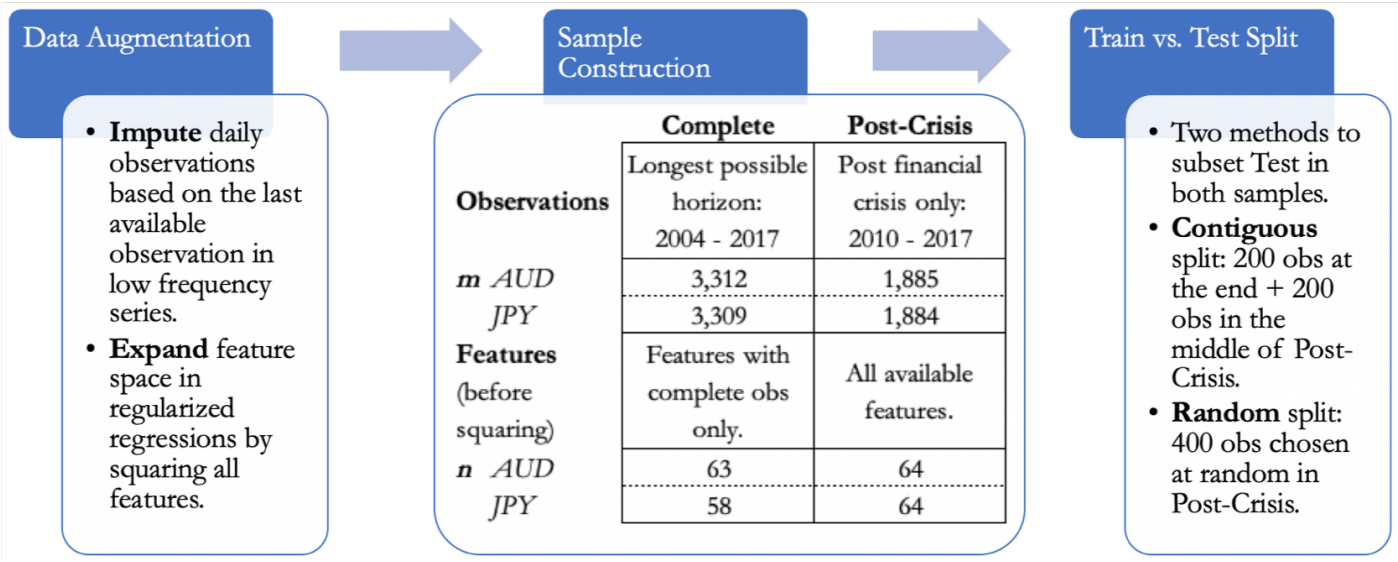
Financial Markets	Economic Conditions	International Trade	Economic Policy
• *15 daily series*	• *7 monthly or quarterly series*	• *5 monthly or quarterly series*	• *5 monthly or quarterly series*
• Stock market indices	• GDP	• Current account balance	• Monetary base
(overall and by sector)	• CPI	• International trade balance	• Broad liquidity
• Stock trading volumes	• Unemployment rate	• Export	• Central bank asset
• 3-month interbank rate	• Labor force participation	• Import	• Government debt
• 10-year government bond yield	• Hourly wage	• International reserve balance	• Economic Policy Uncertainty Index
	• Mortgage outstanding		
	• Loans to bank		

sample focuses on only the years of 2010 to 2017, which is the prediction period of interest. The shorter horizon of *Post-Crisis* allows us to retain all features. The resulting number of features and observations are summarized in Figure 3, along with the complete process of data construction.

Finally, we split each of our two samples into Test vs. Training sets in two different ways. In both cases, we arrive at a test set of 400 observations, which is about a year and half in calendar days. The *Contiguous* split uses as Test set the last 200 observations and the middle 200 observations in the Post-Crisis period; this method emphasizes the time series nature of our outcome variable. The *Random* split uses as Test set 400 randomly chosen observations in the Post-Crisis period, which reflects more of a cross-sectional test of the basis predictions.

Ultimately, each of our ML models is applied to 8 distinct Sample-Split combination: for each of the AUD and JPY bases, we have either the *Complete* or the *Post-Crisis* sample, and within each sample, we split Train vs. Test using either a *Contiguous* or a *Random* approach.

Figure 3: **Data Construction**



3 Methods

3.1 Regularized Regression

Regularized linear regressions extend the ordinary least squares regression algorithm by allowing regularizations on the fitted model parameters. Regularization prevents the linear regression from overfitting, especially when a large set of features are present. Any regularized linear regression solves the following optimization program.

$$\hat{\theta} = \arg\min_{\theta} \sum (y^{(i)} - \theta^T x^{(i)})^2 + \lambda(\alpha||\theta||_2^2 + (1-\alpha)||\theta||_1), \tag{2}$$

where λ is a hyperparameter that determines the "strength" of regularization, and α determines the specific regularization function employed.

Setting $\alpha = 0$ corresponds to regularizating L_1 norm, which is also known as Lasso regression. This regularization will set less relevant θs to zero, thereby achieving model parsimony. An $\alpha = 1$ corresponds to using L_2-regularization, or Ridge regression. It shrinks the coefficients of variables that are highly correlated. Both of these algorithms also have a Bayesian interpretation, with lasso corresponding to a Laplace prior and Ridge to a normal prior over the regression coefficients θ. Finally, we also consider an elastic net algorithm with $\alpha = \frac{1}{2}$. This trades off the two previous regularization methods.

In addition to linear features, we also consider a second-degree polynomial of the features in these regularized regressions in order to capture non-linearities. All features and the y are standardized when training the algorithm in order for the regularization to work as intended. We implement all algorithms in the statistical software R (R Core Team, 2013), and use `glmnet` package (Friedman et al., 2010) to implement these regularized regressions.

3.2 Regression Trees

An algorithm might need to capture higher dimensional feature interactions in order to predict well. A regression tree allows for considering such non-linear interactions among features. In each step, the tree splits the data in one node (the parent node) into two subsets (children nodes) based on the value of one feature. The splitting rule is chosen to minimize the purity of the children nodes. The algorithm stops once the purity of the children nodes does not improve over the purity of the parent node. The prediction in each node j that contains m_j training examples is μ_j:

$$\mu_j = \frac{1}{m_j} \sum_{i \in j} y^{(i)}. \tag{3}$$

And the purity of a node D_j is calculated with the deviance measure:

$$D_j = \frac{1}{m_j} \sum_{i \in j} (y^{(i)} - \mu_j)^2. \tag{4}$$

We implement the regression tree algorithm using the R-package `tree` (Ripley, 2018).

3.3 Random Forest

Regression tree algorithms can exhibit high variance. This problem can be remedied using a random forest. The random forest uses bootstrap to grow multiple trees and returns the average prediction of those trees as its final prediction. It can be shown that the variance of a random forest containing N trees, each with variance σ^2 and correlation ρ is:

$$\rho\sigma^2 + \frac{1-\rho}{N}\sigma^2. \tag{5}$$

Hence, the overall variance can be decreased by choosing a high number of trees N (we choose to grow 2000), and decorrelating the trees to achieve a low ρ. The forest decorrelates its trees in two ways. First, each tree is grown out of a bootstrapped sample, which is different for each tree. Moreover, at each node, the algorithm only considers splitting on a random sub-sample of all available features. The size of this sub-sample is *mtry*, and is a hyperparameter that we will tune. Each tree in the forest grows until it reaches a minimum number of terminal nodes/leaves, and that is set to five in our case. These measures contribute to less correlated trees and a lower overall variance of the random forest. We implement the random forest algorithm using the R-package `ranger` (Wright and Ziegler, 2017).

4 Results and Discussions

Before presenting the ML results, we discuss the tuning of two key hyper-parameters: λ in regularized regressions, and *mtry* in random forest.

Across Lasso, Ridge, and Elastic Net, we tune λ, the strength of our regularization penalty via the one-standard-error approach. That is, the `glmnet` package supplies 100 λ's, we choose the optimal λ as the value that's one standard error away from the λ that minimizes the cross-validation error, so as to reflect estimation errors inherent in calculating MSEs (Friedman et al. (2001)). We do ten-fold cross-validation, and use the same assignment ID across all regularized regressions.

Random Forest has many degrees of freedom. We focus on optimizing *mtry*, or the number of randomly chosen features a node could split on. Splitting only on a subset of features at each node reduces correlation among trees and drives down variance of the overall model. For each of the 8 random forests (one on each Sample-Split), we ran `ranger` with $mtry = \{5, 6, 7, ..., 15\}$. The heuristic is to set *mtry* equal to the square root of feature dimension, which would be 7 or 8 depending on the Sample-Split. We choose the optimal *mtry* to be the number that minimizes the out-of-bag MSE in the Training set. Our final *mtry*'s include four 12's, four 14's, one 10, and one 15. We set the other parameters of the Forest to 2000 trees and minimum of 5 leaves per node.

The performance of all eight of our ML models on the four *Contiguous* splits are summarized in Figure 4. We measure accuracy of prediction by MSE on the Test set. For these *Contiguous* splits, we have two blocks of Test sets: one in the middle of the Post-Crisis period and one at the end. The MSE for a model on this split is taken to be the average MSE in both blocks.

The MSE on the *Random* splits are not shown due to space constraint. The performance on the *Random* splits are stunningly good: the Random Forests generate MSEs on the Test set of between 10 to 20, which is incredibly small compared to the 10 to 25 standard deviation of the outcome variables. However, we embrace this success with reservation as it is difficult to interpret randomly selected observations with imputed values.

Focusing our analysis on the performance on the *Contiguous* splits, we highlight three takeaways.

Figure 4: **Summary of MSEs from Models on Contiguous Split**

	Tree	Forest	Lasso		Ridge		Elastic Net	
			Linear	Polynomial	Linear	Polynomial	Linear	Polynomial
Test MSE								
AUD Complete	222	***69***	655	282	1231	189	726	208
AUD Post-Crisis	333	***70***	247	115	140	154	238	101
JPY Complete	1234	***274***	394	555	314	295	376	527
JPY Post-Crisis	***187***	337	908	1162	820	678	899	1049
Train MSE								
AUD Complete	130	75	242	127	255	120	240	126
AUD Post-Crisis	27	21	33	21	37	38	33	21
JPY Complete	217	84	367	119	405	110	364	121
JPY Post-Crisis	51	14	55	24	67	70	56	24

Note: for algorithms trained on the Contiguous Split, the reported MSE is the average between the MSEs on the two contiguous test blocks.

Legend | ***Lowest*** | Graphed

First, random forests achieve strong prediction performance. Forests not only have the lowest MSE in Test sets in all but one sample, but do so with a substantial margin. Comparing to regularized regressions, forests allow non-linear effects, and compare to regression trees, forests lower variance by bagging and splitting on only a subset of features at each node. The superior performance of forests suggest that these are two important considerations.

We plot in Figure 5 our preferred specification: the forest application to predicting the AUD basis in the Complete sample. We further explore the importance of each feature in this forest. Specifically, in each of the 2000 trees, we tally the first seven variables that the tree split on. The bar height in Figure 6 corresponds to the number of times a feature was actually included in the "Top 7" shortlist. The horizontal line represent the number of times we would expect to see a feature appear if all of the features are equally important and the selection is a random draw from a multinomial distribution with $p_1 = p_2 = ... = p_n = \frac{1}{n}$. Under this heuristic, the features colored in blue are the more important ones. Interestingly, this set of features encompass all but one feature that the regression tree used for the same sample.

Second, regularized regressions are informative about bias vs. variance in the prediction. Looking at the MSEs in the Training vs. Test sets across the linear vs. polynomial specifications of regularized regressions, we note that the prediction error in the AUD basis is likely caused by a bias problem, as the MSE decreases with the inclusion of the higher-dimensional features in both the Training and the Test set. In contrast, the results in JPY basis indicates a variance problem, i.e. overfitting, as the polynomial improves the Training error but increases the Test error.

Finally, performance differ dramatically in the middle vs end Test blocks. In results not shown due to space constraint, we note that all models have respectable performance on the Test block taken from the middle of the Post-Crisis period. Yet most models struggle with predictions in the last 200 observations. One potential reason is that outcome variables in this period exhibit patterns that have hitherto not been observed (low variance, elevated level), and are thus difficult to predict via a supervised learning algorithm.

5 Conclusions and Future Work

Violations of the Covered Interest-Rate Parity condition are important phenomena in the global foreign exchange market, and a better understanding of the cross-currency basis can have profound implications on the theory of asset pricing. In this project, we take a step toward this understanding by predicting bases using machine learning techniques. We find that random forests achieve fairly good predictions, as measured by MSE on Test sets that encompass two separate blocks of observations in the Post-Crisis period. This performance likely owes to random forest's ability to flexibly introduce non-linear feature effects and strike a balance between bias and variance minimization.

In the future, we would collect more economic features and use higher order polynomial features to improve the regularized linear regressions of AUD basis, given the observed bias issue with the AUD data. We will expand the set of algorithms employed to improve the performance on the JPY data, as most algorithms seem to suffer from a variance problem. Specifically, we will apply the boosting technique, and we will consider training a neural network.

Overall, we are encouraged to see that we found models that perform reasonably well. Importantly, the features selected as important by our various models are intuitive and sensible. We hope to more closely examine the contribution of these features in the future and extend this analysis to a larger set of currency bases.

Figure 5: **Outcome vs. Prediction in Training vs. Test Sets**

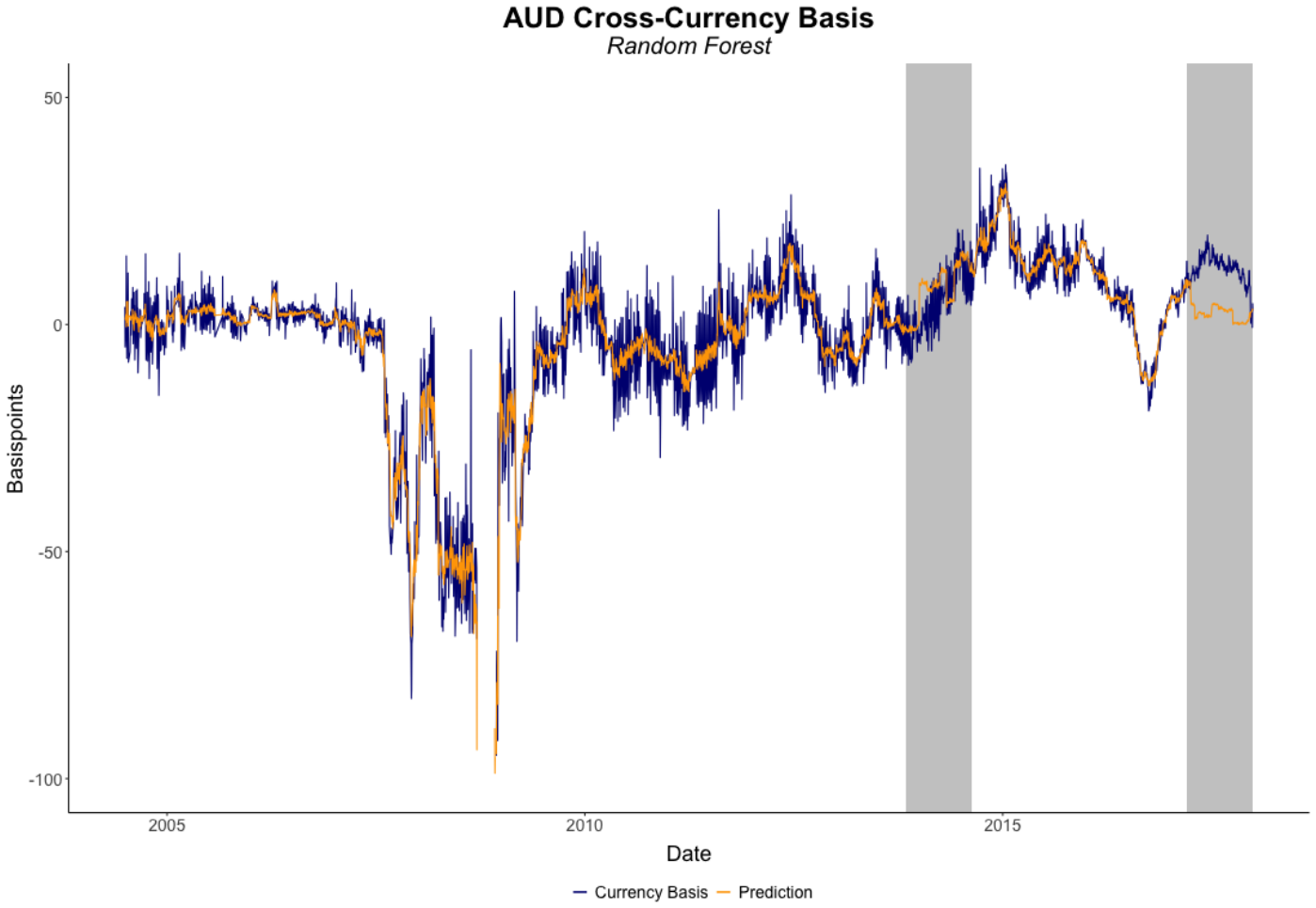
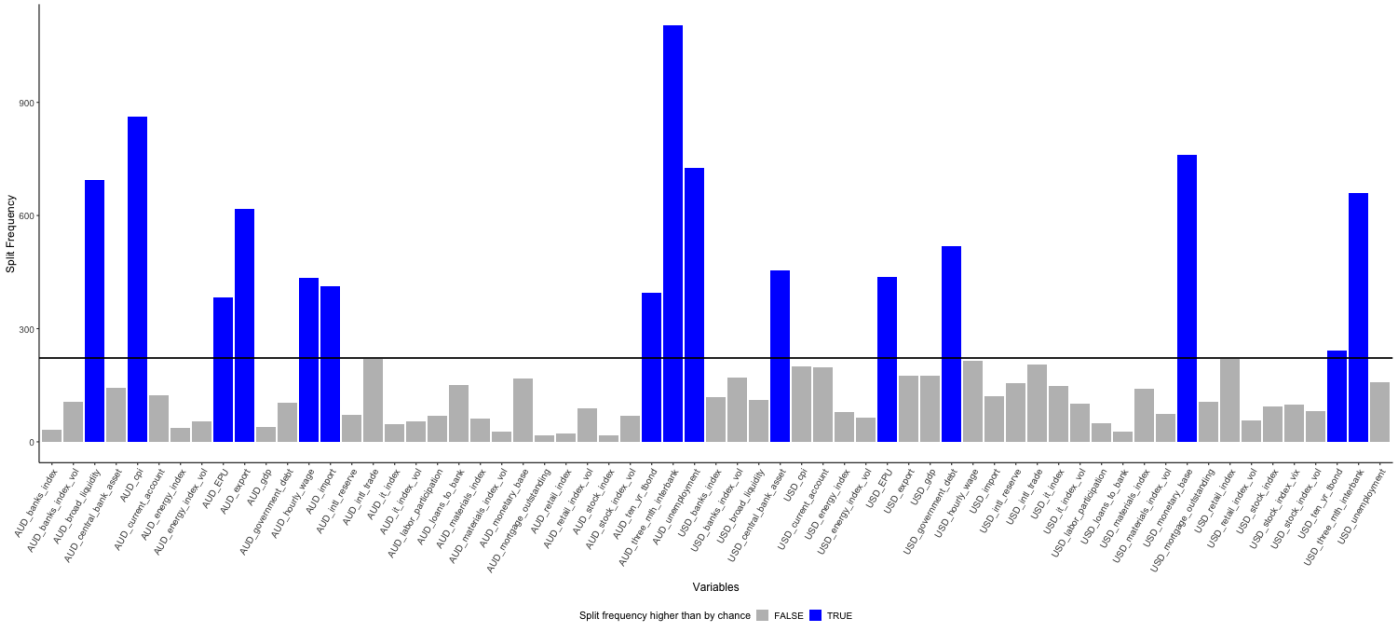


Figure 6: **Variable Importance in Random Forests on AUD Complete Sample Contiguous Split**



6 Contributions

All tasks were performed by Amy and Stefan in equal parts.

References

Boyarchenko, N., Eisenbach, T. M., Gupta, P., Shachar, O., and Van Tassel, P. (2018). Bank-intermediated arbitrage.

Du, W., Tepper, A., and Verdelhan, A. (2018). Deviations from Covered Interest Rate Parity. *The Journal of Finance*.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York, NY, USA:.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

Gu, S., Kelly, B. T., and Xiu, D. (2018). Empirical asset pricing via machine learning.

Hébert, B. and Wang, A. (2018). Forward Arbitrage and Intermediary Asset Pricing.

Kozak, S., Nagel, S., and Santosh, S. (2019). Shrinking the cross section. *The Journal of Financial Economics*, forthcoming.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Ripley, B. (2018). *tree: Classification and Regression Trees*. R package version 1.0-39.

Wright, M. N. and Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77(1):1–17.