1-2018

Prediction Intervals for Functional Data

Nicholas Rios
Montclair State University

ABSTRACT

Title of Thesis: PREDICTION INTERVALS FOR FUNCTIONAL DATA

Nicholas Rios, Master of Science, 2018

Thesis directed by: Dr. Andrada E. Ivanescu

Department of Mathematical Sciences

The prediction of functional data samples has been the focus of several functional data analysis endeavors. This work describes the use of dynamic function-on-function regression for dynamic prediction of the future trajectory as well as the construction of dynamic prediction intervals for functional data. The overall goals of this thesis are to assess the efficacy of Dynamic Penalized Function-on-Function Regression (DPFFR) and to compare DPFFR prediction intervals with those of other dynamic prediction methods. To make these comparisons, metrics are used that measure prediction error, prediction interval width, and prediction interval coverage. Simulations and applications to financial stock data from Microsoft and IBM illustrate the usefulness of the dynamic functional prediction methods. The analysis reveals that DPFFR prediction intervals perform well when compared to those of other dynamic prediction methods in terms of the metrics considered in this paper.

MONTCLAIR STATE UNIVERSITY

PREDICTION INTERVALS FOR FUNCTIONAL DATA

by

Nicholas Rios

A Master's Thesis Submitted to the Faculty of

Montclair State University

In Partial Fulfillment of the Requirements

For the Degree of

Statistics – Master of Science

January 2018

College/School Science and Mathematics

Department Mathematical Sciences

Thesis Committee:

[redacted]

Dr. Andrada E. Ivanescu

Thesis Sponsor

[redacted]

Dr. Andrew McDougall

Committee Member

[redacted]

Dr. Helen M. Roberts

Committee Member

PREDICTION INTERVALS FOR FUNCTIONAL DATA

A THESIS

Submitted in partial fulfillment of the requirements

for the degree of Master of Science in Statistics

by

NICHOLAS RIOS

Montclair State University

Montclair, NJ

2018

Contents

List of Figures

Prediction Intervals for Functional Data

1. Introduction to Functional Data

Functional data are realizations of functions that are observed over a continuum, such as time. We assume that the observed data are generated by some underlying stochastic process, and we can observe the data over a discrete set of time points (Ramsay and Silverman, 2005). Various methods exist for analyzing functional data over a given set of time points. In formal notation, each curve is denoted as $Y_i(t)$, with $i = 1,2, \dots, n$. Here, t is a time point, with $t = 1,2, \dots, M$. As an example, an individual curve could represent monthly stock highs for a year. In this case, $Y_i(t)$ represents the stock high for month t for year i. For instance, $Y_1(2)$ would indicate the monthly stock high for month 2 (February) for year 1 in the dataset.

Another example of functional data is the Canadian Weather dataset. This dataset is provided in R's FDA package (Ramsay and Silverman, 2005; Ramsay et al., 2017). Daily temperatures (in degrees Celsius) were recorded at 35 locations in Canada over the years 1960 to 1994. An average daily temperature was then calculated for each day of the year. In this example, $Y_i(t)$ represents the average daily temperature for day t at location i. Here, $t = 1,2, \dots, 365$ and $i = 1,2, \dots, 35$. This dataset also contains information on the average daily rainfall for each day of the year at each of the 35 locations.

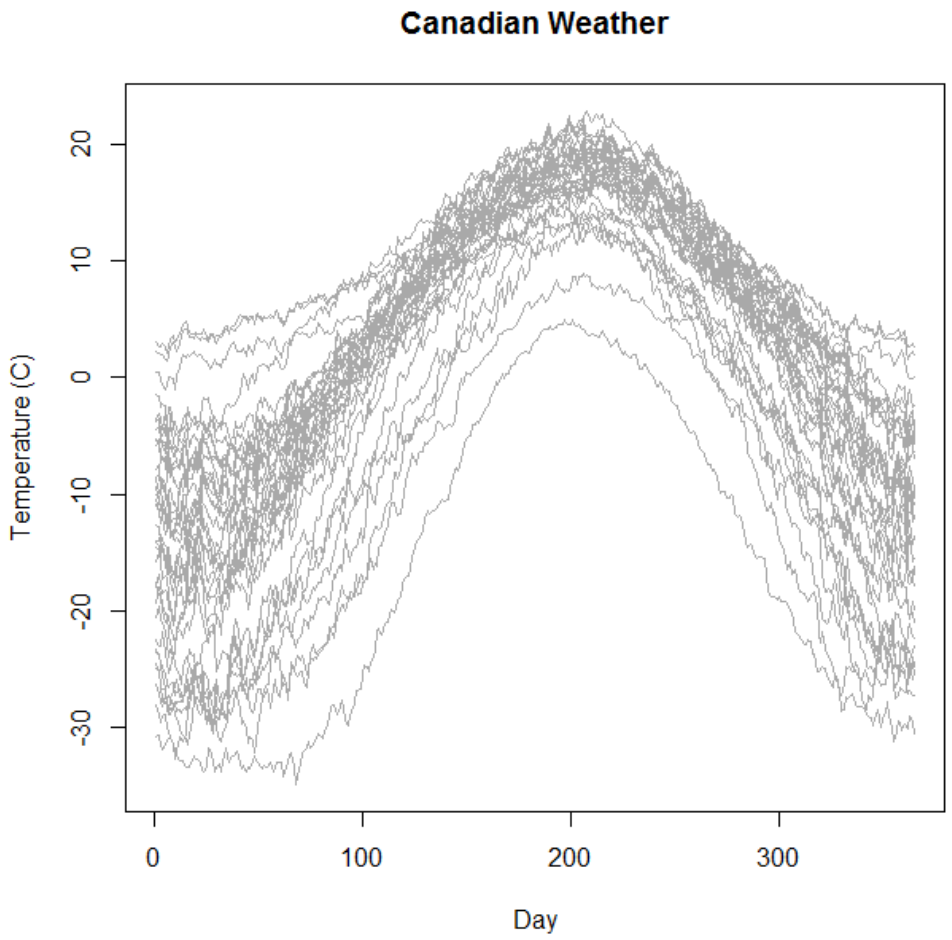

Figure 1. Depicted are the functional data curves for average daily temperatures for n=35 locations in Canada.

Figure 1 shows the average daily temperature for each day of the year for each of the 35 locations in Canada. It can be observed that the average daily temperature increases towards the middle of the year, and then decreases at the end of the year.

1.1. Dynamic Prediction

The focus of this thesis is dynamic prediction, and specifically, prediction intervals used for dynamic prediction. Methods of dynamic functional prediction have a primary focus to predict the trajectory in the future based on known historical data. Formally, we

can call the cutoff point r, and say that we are predicting the trajectory of a curve $Y_i(\tilde{t})$ for $\tilde{t} \in \{r+1, r+2, \dots, M\}$, based on the historical data on the curves $Y_i(t)$ for $t \in \{1,2, \dots, r\}$. Once the prediction for a specific curve is obtained, upper and lower bounds for the predicted trajectory of that curve can be computed to make a prediction interval. All analyses for this thesis were run in R (R Core Team, 2016).

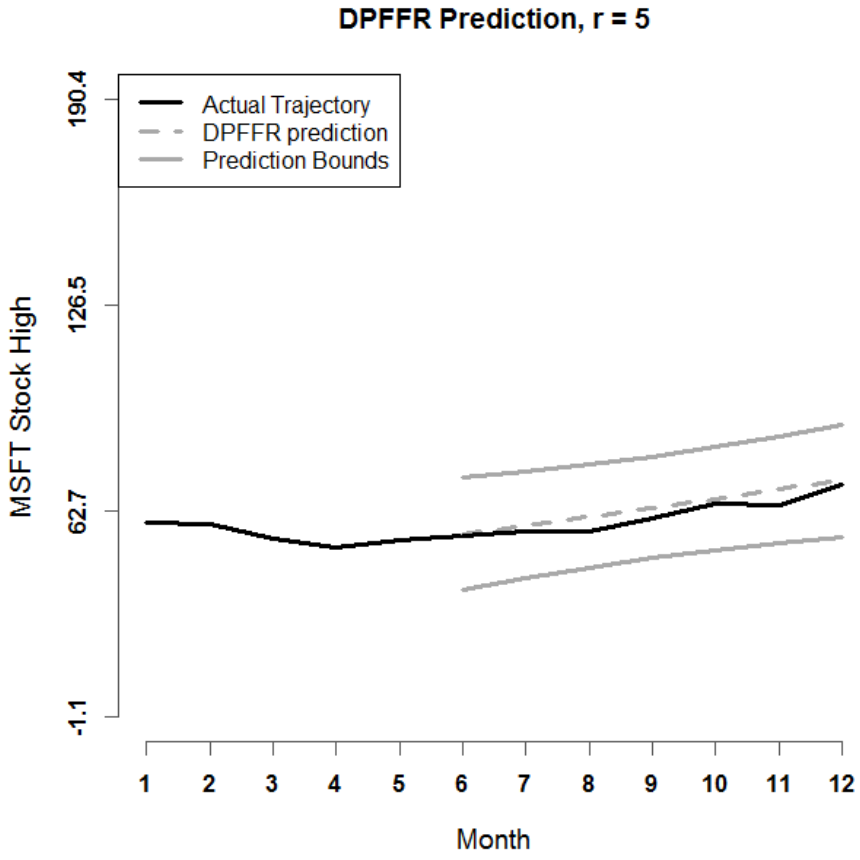

Figure 2. Depicted above is a prediction interval for the Microsoft (MSFT) stock highs in a fixed year, r = 5.

As an illustrated example of dynamic prediction, Figure 2 displays a dynamic prediction and a prediction interval of the stock highs for Microsoft for a fixed year (Microsoft Historical Prices, 2016), with cutoff point r = 5. In this example, the stock high

is the highest price that a stock obtains in a given month. The time points are months $t = 1,2, \ldots ,12$, and the original dataset has n = 29 curves, with each curve corresponding to one year from 1987 to 2015. The true trajectory of the stock highs is shown as a solid black curve, the dynamic prediction is shown as a dashed gray curve, and the prediction interval bounds are depicted as solid gray curves. The dynamic prediction method being used here is using historical data from the curves up to the month of May (r = 5) to generate the dynamic prediction of the stock highs for each month after May. It can be observed that this prediction interval contains the true trajectory of the stock highs for this year.

There are already several methods in the functional data analysis literature that are used for dynamic prediction. One is called dynamic functional linear regression (Shang, 2015, Section 4.6), implemented in software as the function dynamic_FLR in the ftsa (Hyndman and Shang, 2017) package in R. It only uses the historical data on the samples of curves of interest and cannot include other predictors. This method extracts the functional principal components of the curves, and conditions on the estimated eigenfunctions to obtain the dynamic forecast based on a dynamic functional regression model (Hyndman and Shang, 2017; Shang, 2015; Shang and Hyndman, 2016). Another method uses penalized least squares (Shang, 2015, Section 4.4) for dynamic prediction, where the estimated eigenfunctions are also used as predictive information. The penalized least squares method is available in R as the dynupdate function of the ftsa R package.

1.2. Literature Review

A large volume of research already exists on the topic of prediction for functional data analysis. A method for estimating functional data curves while accounting for uncertainty in functional principal components analysis was developed (Goldsmith et al., 2013). Predictions from scalar-on-function regression were proposed based on cross validation and ensemble prediction (Goldsmith and Scheipl, 2014). A semi-functional partial linear regression model was developed to predict future observations (Aneiros-Pérez and Vieu, 2008). Several methods of creating simultaneous prediction intervals were compared and analyzed in the context of a French electricity consumption function-valued dataset (Antoniadis et al., 2016). These methods demonstrate that there are a wide range of methods for analyzing functional data, however, dynamic prediction is not discussed. These papers also indicate that there might be a variety of applications in need of analysis that involve prediction in functional data scenarios. A good example of a functional data scenario that involves prediction is spectrometric data. Data gathered from a Teactor Infrared Spectrometer was used to predict the fat content of meat using absorbance as a predictor in the context of functional regression (Goldsmith and Scheipl, 2014). Another application of prediction in functional data analysis is atmospheric ozone concentration. Regression for functional data was used in an application to predict future ozone concentrations based on historical functional data on ozone, nitrogen dioxide, and sulfur dioxide levels (Aneiros-Pérez and Vieu, 2008). In this application, chemical measurements of ozone, nitrogen dioxide, and sulfur dioxide were taken every hour over 124 days in 2005 in Madrid, Spain, and these chemical measurements were used as predictors of future ozone concentration in 7 non-functional regression models and 7 regression models that are classified as either functional or semi-functional (Aneiros-Pérez and Vieu, 2008).

2. Research Goals

A main goal of this thesis is to assess the efficacy of the dynamic predictions and dynamic prediction intervals produced by Dynamic Penalized Function-on-Function Regression (DPFFR). DPFFR will be compared with other dynamic prediction methods (BEnchmark DYnamic, Dynamic Linear Model, Dynamic Penalized Functional Regression, Dynamic Functional Linear Regression, and Penalized Least Squares) in terms of prediction error, prediction interval width, and prediction interval coverage. These metrics will be evaluated for several different simulated datasets. In addition, this research intends to apply these dynamic prediction algorithms to real financial stock data and see the prediction performance when applied to real data. R software is used for the implementations of methods for dynamic prediction intervals.

The next section describes the six dynamic prediction methods that will be used to construct dynamic predictions and prediction intervals. Section 4 describes the metrics that will be used to compare the dynamic predictions and prediction intervals. Section 5 explains how functional data curves were simulated, and it also provides a numerical analysis of the performance of the six dynamic prediction methods in simulations. Section 6 details the application of three dynamic prediction methods (DPFFR, Dynamic Functional Linear Regression, Penalized Least Squares) to real financial stock data and provides the resulting analysis.

3. Comparison of Methods for Dynamic Prediction

There are a variety of methods that have been developed for dynamic prediction, in the context of functional datasets. All prediction methods listed in this paper are implemented using leave-one-out cross validation. When predicting the future trajectory of a curve, that curve's historical functional data is left out of the data set used to generate the model fit used for predictions. So, when predicting $Y_i(\tilde{t})$ (for fixed i), the corresponding historical functional data $Y_i(t)$ is excluded when obtaining the model fit. Methods that will be discussed and compared in this paper include DPFFR, BENDY, DLM, DPFR, and the methods of Shang (2016) and Hyndman and Shang (2017) denoted as dynamic_FLR and dynupdate, respectively.

3.1. Dynamic Penalized Function-on-Function Regression (DPFFR)

A main goal of this thesis is to study Dynamic Penalized Function-on-Function Regression (DPFFR) and compare the prediction intervals of DPFFR with those of other methods. DPFFR makes predictions based on historical data from the curves of interest and available useful covariates. DPFFR is used to make dynamic predictions within penalized regression (Wood, 2006) in the framework of function-on-function regression (Ivanescu et al., 2015). The model we study is given below.

$$Y_i(\tilde{t}) = W_{i1}\gamma_1 + W_{i2}\gamma_2 + \zeta(\tilde{t}) + \int_{\mathrm{T}} Y_i(t)\beta(\tilde{\mathrm{t}}, \mathrm{t})\, dt + \int_{\mathrm{T}} Z_i(t)\delta(\tilde{\mathrm{t}}, \mathrm{t})dt + \epsilon_i(\tilde{\mathrm{t}}). \qquad (1)$$

Here, the $Y_i(\tilde{t})$ are the functional responses over time domain $\tilde{t} \in \{r+1, r+2, \dots, M\}$, which is the future time domain where we want to make predictions. Scalar covariates W_{i1}, W_{i2} have scalar effects on the predicted responses given by coefficients γ_1 and γ_2, respectively. The model term $\zeta(\tilde{t})$ is the functional intercept. Functional predictors $Y_i(t)$ and $Z_i(t)$ are the historical curves of interest and the functional covariate,

respectively. Both are observed at time points $t \in \{1,2,\ldots,r\}$. Model parameters $\beta(\tilde{t},t)$ and $\delta(\tilde{t},t)$ are bivariate functional parameters that are larger in magnitude when the data they are paired with at time t is useful for making predictions at time $\tilde{t}$. The term $\epsilon_i(\tilde{t})$ is an error term. The integrals in the model are taken are over the domain $t \in \{1,2,\ldots,r\}$.

For example, $Y_i(\tilde{t})$ could be the energy consumption of a city for the second half of the year, with measurements taken daily. Then $Y_i(t)$ would be the available daily energy consumption for the first half of the year that can be used to obtain dynamic predictions for $Y_i(\tilde{t})$. Moreover, $Z_i(t)$ may be the temperature recorded for the first half of the year, measured daily, as temperature would be a useful functional predictor for energy consumption. Parameter $\beta(\tilde{t},t)$ would be larger when the energy consumption in the past is useful for making predictions in the future, and likewise, $\delta(\tilde{t},t)$ would be larger when the temperature in the first half of the year is useful for predicting the energy consumption for the future part of the year.

The goals of DPFFR are:

1. To fit the DPFFR model to functional data and estimate the model parameters.
2. To obtain predictions $\hat{Y}_i(\tilde{t})$ as an estimate for the future trajectory $Y_i(\tilde{t})$.
3. To obtain prediction intervals, upper and lower bounds.

Model (1) is a function-on-function regression model for dynamic prediction, where the response variable is a functional dataset. Functional predictors are used to predict the functional responses in a dynamic functional regression framework. The model parameters are estimated by applying a penalized least squares criterion with the aim of obtaining estimators for the smooth functional parameters that minimize the criterion. The

penalization ensures that the resulting functional parameter estimates are smooth functional estimators. Once these model parameters are estimated, they can be used to make predictions for $Y_i(\tilde{t})$ and compute the estimated variability associated with $Y_i(\tilde{t})$, which lets us construct prediction intervals.

More specifically, we employ functional regression that uses smoothing penalties in a functional regression approach based on penalized splines. Parameters $\zeta(\tilde{t}), \beta(\tilde{t}, t)$, and $\delta(\tilde{t}, t)$ are assumed to be functional. We choose a large number of splines basis functions to represent the DPFFR functional parameters and apply the penalties $\lambda_\zeta P(\zeta)$, $\lambda_\beta P(\beta)$, and $\lambda_\delta P(\delta)$. If we denote by $f_{i,\tilde{t}}(\gamma_1, \gamma_2, \zeta, \beta, \delta)$ the mean of the $Y_i(\tilde{t})$, the penalized criterion to be minimized is shown below

$$\sum_{i,\tilde{t}} \left|\left| Y_i(\tilde{t}) - f_{i,\tilde{t}}(\gamma_1, \gamma_2, \zeta, \beta, \delta) \right|\right|^2 + \lambda_\zeta P(\zeta) + \lambda_\beta P(\beta) + \lambda_\delta P(\delta).$$

This is a penalized least squares criterion and corresponds to a residual sum of squares criterion specific for the setting of dynamic penalized functional regression. The penalized criterion is in alignment with popular penalization techniques for functional data, such as the criterion in Goldsmith et al., (2011, 2012).

3.2. BENchmark DYnamic (BENDY)

BENDY is the BENchmark DYnamic method of dynamic prediction. This model predicts the individual response $Y_i(\tilde{t})$ in the future time domain $\tilde{t}$ by using the first and last observations $Y_i(1)$ and $Y_i(r)$ from the historical data curves and the first and last observations $Z_i(1)$ and $Z_i(r)$ from a functional covariate. It can also account for several scalar covariates. In this method, the following model is considered:

$$Y_i(\tilde{t}) = W_{i1}\gamma_1 + W_{i2}\gamma_2 + \zeta(\tilde{t}) + Y_i(1)\beta(\tilde{t},1) + Y_i(r)\beta(\tilde{t},r) + Z_i(1)\delta(\tilde{t},1) + Z_i(r)\delta(\tilde{t},r) + \epsilon_i(\tilde{t}). \quad (2)$$

In model (2), $Y_i(\tilde{t})$ is taken as a scalar response that is being predicted by the BENDY method at each point from the grid of $\tilde{t}$. This makes BENDY different from DPFFR, since BENDY is used to predict a scalar response at each $\tilde{t}$ in turn, while DPFFR is used to predict a functional response (i.e. the entire future curve having domain $\tilde{t}$). The W_{i1} and W_{i2} terms are scalar covariates, $\zeta(\tilde{t})$ is an intercept term, and $\epsilon_i(\tilde{t})$ is an error term. Observations from $Y_i(t)$ and $Z_i(t)$ are taken as covariates in the BENDY model. For instance, $Y_i(1)$ is the first observation from the historical data curves, and likewise, $Y_i(r)$ is the r^{th} (the last) observation from the historical data curves. Similarly, $Z_i(1)$ and $Z_i(r)$ are the first and r^{th} observations from the covariates. BENDY differs from other dynamic prediction models because it only uses the first and the r^{th} points from the historical data and covariates to make predictions, whereas other dynamic prediction methods (such as DLM, DPFR, and DPFFR) use all of the historical data from time points $1,2,\dots,r$ available to make predictions.

In R, the linear model (lm) method is used to fit the BENDY model at each time point $\tilde{t}$ in turn. The resulting model fit, together with the predictive information from the i^{th}sample are used to find prediction intervals with confidence level $C = 100(1-\alpha)\%$ of the following form:

$$\hat{Y}_i(\tilde{t}) \pm t^*_{df_{MSE}} * \sqrt{Var\{\hat{Y}_i(\tilde{t})\} + Var\{\epsilon_i(\tilde{t})\}},$$

where $\hat{Y}_i(\tilde{t})$ is the predicted value from the BENDY model, $Var\{\hat{Y}_i(\tilde{t})\}$ is the variance of the prediction at time point $\tilde{t} \in \{r+1,\dots,M\}$, and $Var\{\epsilon_i(\tilde{t})\}$ is the variance of the error.

The critical value is $t^*_{df_{MSE}}$, which is a quantile of the t-distribution with the same degrees of freedom as the mean squared error, MSE, and corresponds to the confidence level C. Note that the BENDY method uses data on all curves except the left-out curve to fit the BENDY linear model, for a total of n-1 curves. In practice, $Var\{\epsilon_i(\tilde{t})\}$ is estimated as shown below

$$MSE = \frac{1}{df_{MSE}} \sum_{i=1}^{n-1} e_i^2,$$

where $e_i^2 = \left(\hat{Y}_i(\tilde{t}) - Y_i(\tilde{t})\right)^2$ is the square of the i^{th} sample residual difference between the fitted BENDY model and the observed data Y_i at time point $\tilde{t}$. The degrees of freedom for the MSE is given by df_{MSE}, which is given as the number of curves minus the number of model parameters (Kutner et al., 2004, Chapter 6). The variance of the predictions is estimated by taking into account the estimated variance of the model parameters in a BENDY model.

Let X be the design matrix that contains the historical data on all n curves. Then X has the form:

$$X = \begin{bmatrix} 1 & W_{11} & W_{12} & Y_1(1) & Y_1(r) & Z_1(1) & Z_1(r) \\ 1 & W_{21} & W_{22} & Y_2(1) & Y_2(r) & Z_2(1) & Z_2(r) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & W_{n-1,1} & W_{n-1,2} & Y_{n-1}(1) & Y_{n-1}(r) & Z_{n-1}(1) & Z_{n-1}(r) \\ 1 & W_{n,1} & W_{n-1,2} & Y_n(1) & Y_n(r) & Z_n(1) & Z_n(r) \end{bmatrix}.$$

Then the design matrix used for the prediction is X_{-i}, where row i corresponds to the historical data on the i^{th} curve that is being predicted. The design matrix X_{-i} will have $n - 1$ rows.

Let the matrix X_i be the prediction matrix containing the historical data (up to time point r) from the i^{th} curve that pertains to the predicted response $\hat{Y}_i(\tilde{t})$, and let $X_i{}'$ denote

the transpose of X_i. The prediction matrix for the i^{th} curve is a 1-row vector that takes the following form:

$$X_i = [1\ W_{i1}\ W_{i2}\ \ Y_i(1) \quad Y_i(r) \quad Z_i(1) \quad Z_i(r)].$$

Then it follows that

$$\widehat{Var}\{\hat{\beta}\} = MSE * (X'_{-i} * X_{-i})^{-1},$$

$$\widehat{Var}\{\hat{Y}_i(\tilde{t})\} = X_i * \widehat{Var}\{\hat{\beta}\} * X'_i.$$

BENDY is implemented in R by using the linear model (lm) routine for model fitting. Prediction intervals and predictions are obtained with the predict.lm function in R.

3.3. Dynamic Linear Model (DLM)

DLM stands for Dynamic Linear Model. This model is similar to BENDY in the sense that DLM also predicts a scalar response instead of a functional response, but DLM uses all of the data points of interest from the historic data to make its predictions. The model for DLM is as follows:

$$Y_i(\tilde{t}) = W_{i1}\gamma_1 + W_{i2}\gamma_2 + \zeta(\tilde{t}) + \sum_{t=1}^{r} Y_i(t)\,\beta(\tilde{t},t) + \sum_{t=1}^{r} Z_i(t)\delta(\tilde{t},t) + \epsilon_i(\tilde{t}). \quad (3)$$

DLM treats $Y_i(\tilde{t})$ as a scalar response, W_{i1} and W_{i2} are scalar covariates, $\zeta(\tilde{t})$ is an intercept, and $\epsilon_i(\tilde{t})$ is an error term. $\beta(\tilde{t},t)$ is a parameter that assumes larger values when the historical data Y at time t is more useful for predicting the response. Likewise, the parameter $\delta(\tilde{t},t)$ would be larger when the Z covariate data at time t is more useful for predicting the response. Both $\beta(\tilde{t},t)$ and $\delta(\tilde{t},t)$ are estimated for each pair $(\tilde{t},t)$.

In a similar fashion to DPFFR, DLM incorporates all of the historical data from $t = 1$ to $t = r$ and other covariates to make predictions, but DLM differs from DPFFR because instead of integrating $Y_i(t)\beta(\tilde{t},t)$ over the domain, DLM sums them over a

discrete set of points to get a scalar response. Furthermore, in DPFFR, $\beta(\tilde{t}, t)$ and $\delta(\tilde{t}, t)$ are taken to be smooth surfaces, but the DLM model requires no such assumption. It should also be noted that DLM uses functional covariates $Z_i(t)$ in its model as well, making it similar to BENDY and DPFFR in this regard. In R, the lm method is used to fit the DLM model. This can be applied to create prediction intervals by using a similar methodology as presented in the BENDY model while talking into account the extended DLM model.

3.4. Dynamic Penalized Functional Regression (DPFR)

DPFR represents the method of Dynamic Penalized Functional Regression. The model for DPFR is the same as model (1) for DPFFR, with the exception that the response $Y_i(\tilde{t})$ is taken to be a scalar, not a function. So, to generate the whole predicted trajectory of a curve, DPFFR only needs to be run once, whereas DPFR would need to be executed one time for each point in the future time domain $\tilde{t}$ in order to construct the entire curve. This implies that the parameters $\beta(\tilde{t}, t)$ and $\delta(\tilde{t}, t)$ would be re-fitted for each time point of $\tilde{t}$ by using methodology for penalized scalar-on-function regression (Goldsmith et al. 2011). The DPFR model integrates the historical functional data over time domain for t, while the DLM model uses a summation over a discrete set of time points $t \in \{1,2, \dots, r\}$. This is because the form of the DPFR functional model parameters is assumed to be a smooth function of t.

3.5. Dynamic Functional Linear Regression (dynamic_FLR)

Dynamic Functional Linear Regression, denoted as dynamic_FLR (Shang, 2015, Section 4.6), is a dynamic prediction method that only uses historical data on the curves of

interest to make predictions; therefore, it does not rely on any covariates. This is an aspect that makes dynamic_FLR different from DPFFR.

This method conditions on the functional principal components of the provided historical data, and uses this information to obtain dynamic predictions (Shang, 2015; Shang and Hyndman, 2016). In a similar fashion to DPFFR, the method can be used when the prediction of $Y_i(\tilde{t})$ is a function or a scalar response. For the numerical implementations in this thesis, the predictions for $Y_i(\tilde{t})$ were requested to be a function, so one execution of dynamic_FLR is sufficient to obtain predictions for the trajectory of an entire future curve. Calling the dynamic_FLR method provided in R's ftsa library (Hyndman and Shang, 2016) will implement this method. The dynamic_FLR method is also used to create dynamic predictions and prediction intervals.

3.6. Penalized Least Squares (Dynupdate)

Another method for dynamic prediction is Penalized Least Squares (Shang, 2015, Section 4.4), which is implemented in R as the dynupdate method of the ftsa package. Dynupdate is similar to dynamic_FLR because both of these methods do not use additional covariates to make predictions, and both methods provide a prediction at the discretized time points for the functional response (similar to DPFFR). Similar to dynamic_FLR, the response $Y_i(\tilde{t})$ may also be taken to be a scalar in other implementations, even though in the implementations we considered the prediction was required to be a function. The model for dynupdate is as follows:

$$\hat{Y}_i(\tilde{t}) = \hat{\mu}(\tilde{t}) + \sum_{k=1}^{K} \hat{\beta}_k^{PLS}\, \hat{\phi}_k(\tilde{t}),$$

where $\hat{\mu}(\tilde{t})$ denotes an estimate of the mean function, $\hat{\phi}_k(\tilde{t})$ is the k^{th} eigenfunction at time point $\tilde{t}$ (according to Functional Principal Components Analysis) corresponding to the $\hat{Y}_i(\tilde{t})$ process, and $\hat{\beta}_k^{PLS}$ is a penalized least squares estimate of the coefficients. See Section 6.1 for more details on Functional Principal Components Analysis. Quadratic penalties are used by dynupdate (Hyndman and Shang, 2017) to estimate the regression coefficients $\hat{\beta}_k^{PLS}$.

The table below summarizes the six dynamic prediction methods that were considered in this thesis.

Table 1. *Summary of dynamic prediction methods.*

Method	Response Type for $Y_i(\tilde{t})$	Covariates in Model	Uses Penalized Criterion to Estimate Parameters
DPFFR	Function	$W_{i1}, W_{i2}, Y_i(t), Z_i(t)$	Yes
BENDY	Scalar	$W_{i1}, W_{i2}, Y_i(1), Y_i(r), Z_i(1), Z_i(r)$	No
DLM	Scalar	$W_{i1}, W_{i2}, Y_i(t), Z_i(t)$	No
DPFR	Scalar	$W_{i1}, W_{i2}, Y_i(t), Z_i(t)$	Yes
dynamic_FLR	Function or Scalar	Uses functional principal components from $Y_i(t)$	No
dynupdate	Function or Scalar	Uses functional principal components from $Y_i(t)$	Yes

4. Metrics

Four metrics were used in this thesis to compare the various methods of dynamic prediction and prediction intervals. They were the Integrated Mean Prediction Error (IMPE), the Average Width (AW), the Average Coverage (AC), and the CPU time. In simulations and the applications, all of these metrics were computed for all considered methods of dynamic prediction and prediction intervals. Once the predicted future trajectory and the prediction interval upper and lower bounds are obtained for all n curves in a given dataset using the leave one-curve out cross-validation technique, we computed the following metrics.

4.1. Integrated Mean Prediction Error (IMPE)

The IMPE is defined as

$$IMPE = \frac{1}{n} * \frac{1}{M-r} \sum_{i=1}^{n} \sum_{\tilde{t}=r+1}^{M} \left(Y_i(\tilde{t}) - \hat{Y}_i(\tilde{t}) \right)^2,$$

where M is the number of time points, r is the cutoff point, and n is the number of curves. The time points for the predicted curve are $\tilde{t} \in \{r+1, r+2, \dots, M\}$, and the summation begins at point $r+1$ and ends at time M. $Y_i(\tilde{t})$ is the observed value of curve i at time point $\tilde{t}$, and $\hat{Y}_i(\tilde{t})$ is the predicted value of curve i at time point $\tilde{t}$. The IMPE is a sum of squared differences between observed and predicted values. The lower the IMPE is, the less error the method has in predicting the trajectories of the future curves. The prediction error is only averaged over the future time domain $\tilde{t}$ because that is the time domain where predictions are made.

4.2. Average Width (AW)

For a given prediction interval method, the mean width (MW) of the dynamic prediction interval corresponding to the i^{th} curve is given below

$$MW_i = \frac{1}{M-r}\sum_{\tilde{t}=r+1}^{M}\left(UB_i(\tilde{t}) - LB_i(\tilde{t})\right),$$

where $\tilde{t} \in \{r+1, r+2, \ldots, M\}$, and r is the cutoff point. $UB_i(\tilde{t})$ is the upper bound of the level C prediction interval for the i^{th} curve at time point $\tilde{t}$. Likewise, $LB_i(\tilde{t})$ is the lower bound of the level C prediction interval for the i^{th} curve at time point $\tilde{t}$. The MW can be thought of as the average of the widths of each prediction interval generated by a dynamic prediction method. To obtain a grand mean width (referred to as the Average Width, or AW) for the entire sample of n curves, one can take the average of the mean widths across all curves. The formula for the AW is given below

$$AW = \frac{1}{n} * \frac{1}{M-r}\sum_{i=1}^{n}\sum_{\tilde{t}=r+1}^{M}\left(UB_i(\tilde{t}) - LB_i(\tilde{t})\right).$$

It is preferred that the AW be consistently small at the nominal level, so that upper and lower prediction bounds will be precise and useful in practice.

4.3. Average Coverage (AC)

The mean coverage based on a dynamic prediction interval for the i^{th} curve is given below

$$MC_i = \frac{1}{M-r}\sum_{\tilde{t}=r+1}^{M} I\{Y_i(\tilde{t}) \in [LB_i(\tilde{t}), UB_i(\tilde{t})]\},$$

where I is the indicator function which is equal to 1 when $Y_i(\tilde{t})$ falls within the level C prediction interval, and is equal to 0 otherwise. The mean coverage will always take on a value between 0 and 1, and it can be interpreted as the proportion of observed values that actually fell between the lower and upper bounds of the prediction interval. The goal of this metric is to evaluate if the prediction intervals are providing appropriate lower and upper bound estimates for the trajectory of the functional curves, given a nominal level. If a method produced prediction intervals that always contained the actual observed value, then the mean coverage would be 1 (indicating 100% coverage). The overall mean coverage (referred to as the Average Coverage, or AC) for a particular method can be determined by taking the average of the mean coverages across all curves in the dataset. The formula for the AC is given below

$$AC = \frac{1}{n} * \frac{1}{M-r} \sum_{i=1}^{n} \sum_{\tilde{t}=r+1}^{M} I\{Y_i(\tilde{t}) \in [LB_i(\tilde{t}), UB_i(\tilde{t})]\}.$$

The AC is the average of the MC, which is calculated for each curve, across all n curves.

4.4. Central Processing Unit Time (CPU)

The fourth metric considered was the Central Processing Unit time. The CPU time is defined as the average amount of time, in seconds, that it takes to execute one iteration of a dynamic prediction method for all n curves. It is generally preferred that the CPU time is small.

5. Numerical Results

R was used to simulate functional data curves. The data are simulated for the context of dynamic functional regression with $M = 16$ time points, $n = \{25, 50\}$ curves, and cutoff points $r = \{8, 11\}$. This resulted in several different simulated functional datasets. Confidence levels $C = \{90,95,99\}$ were considered for the construction of prediction intervals. The method of generating these data sets is detailed below.

5.1. Simulation Design

We begin with a simulation study where the functional responses $Y_i(\tilde{t})$ are generated from two functional predictors and no scalar covariates. The model used is:

$$Y_i(\tilde{t}) = \zeta(\tilde{t}) + \int_{\mathrm{T}} Y_i(t)\beta(\tilde{\mathrm{t}}, \mathrm{t})\, dt + \int_{\mathrm{T}} Z_i(t)\delta(\tilde{\mathrm{t}}, \mathrm{t})dt + \epsilon_i(\tilde{\mathrm{t}}). \quad (4)$$

The data were also simulated according to the DPFFR model (1). The data are first simulated according to model (4), a model with two functional predictors and no scalar covariates, by creating several realizations of functional random variables in R, and using these to generate the functional responses for the model. The bivariate parameter $\beta(\tilde{t}, t)$ was generated using the following formula:

$$\beta(\tilde{t}, t) = \cos\left(\frac{2\tilde{t}\pi}{16}\right)\sin\left(\frac{2t\pi}{16}\right).$$

There are two settings (denoted as setting A and setting B) for the simulation that determine the parameter $\delta(\tilde{t}, t)$.

Setting A. $\delta(\tilde{t}, t) = \frac{\sqrt{t}\sin\left(\frac{2\tilde{t}\pi}{16}\right)}{4.2}$

Setting B. $\delta(\tilde{t}, t) = \frac{\sqrt{t\tilde{t}}}{4.2}$

Setting A uses a combination of a sine and polynomial function to generate $\delta(\tilde{t},t)$, while Setting B uses a combination of polynomials to generate $\delta(\tilde{t},t)$. The functional intercept is $\zeta(\tilde{t}) = e^{-(\tilde{t}-12.5)^2}$. The formulas for $\beta(\tilde{t},t), \delta(\tilde{t},t)$, and $\zeta(\tilde{t})$ are similar to those presented in Ivanescu et al., 2015. The errors $\epsilon_i(\tilde{t})$ were generated as $N(0,0.22^2)$ random variables. A similar variance of the error terms was considered in the literature, see Goldsmith et al., 2013. The historical functional data were taken to be

$$Y_i(t) = \sum_{k=1}^{10}\left[\rho_{i,k_1}\sin\left(\frac{2k\pi t}{10}\right) + \rho_{i,k_2}\cos(2k\pi t)\right],$$

where ρ_{i,k_1} and ρ_{i,k_2} are generated as $N(0,\frac{1}{k^2})$ random variables that are independent across the curves i, where $k = 1,\dots,10$ (Goldsmith et al., 2011). The functional covariates are given by

$$Z_i(t) = \sum_{k=1}^{40}\left(\frac{2\sqrt{2}}{k\pi}\right)U_{i,k}\sin\left(\frac{k\pi t}{16}\right),$$

where $U_{i,k}$ are $N(0,1)$ random variables.

The second round of simulations used model (1),

$$Y_i(\tilde{t}) = W_{i1}\gamma_1 + W_{i2}\gamma_2 + \zeta(\tilde{t}) + \int_{\mathrm{T}} Y_i(t)\beta(\tilde{\mathrm{t}},\mathrm{t})\,dt + \int_{\mathrm{T}} Z_i(t)\delta(\tilde{\mathrm{t}},\mathrm{t})dt + \epsilon_i(\tilde{\mathrm{t}})$$

which includes two scalar covariates W_{i1} and W_{i2}. The scalar covariates were generated as $W_{i1} = I\{U[0,1] \geq 0.75\}$ and $W_{i2} \sim N(0,0.1^2)$, where I is the indicator function that returns 1 if a standard uniform random variable is greater than or equal to 0.75. The corresponding scalar effects were simulated as $\gamma_1 = 1, \gamma_2 = -0.5$ (Ivanescu et al., 2015).

In Figures 3 and 4, two sets of n = 50 curves are depicted in gray lines that were simulated using the methods previously discussed. Three of the curves have been highlighted to illustrate samples of simulated dynamic functional samples. In Figure 3, the

historical curves $Y_i(t)$ have been generated starting from time point 0 up to time point r = 8. Then, model (4) was used to generate the curves from time points $r + 1$ to 15, using the simulated $Y_i(t)$ and other model components as outlined in model (4), for the time points beyond r = 8. Figure 4 illustrates simulated data for the case when r = 11.

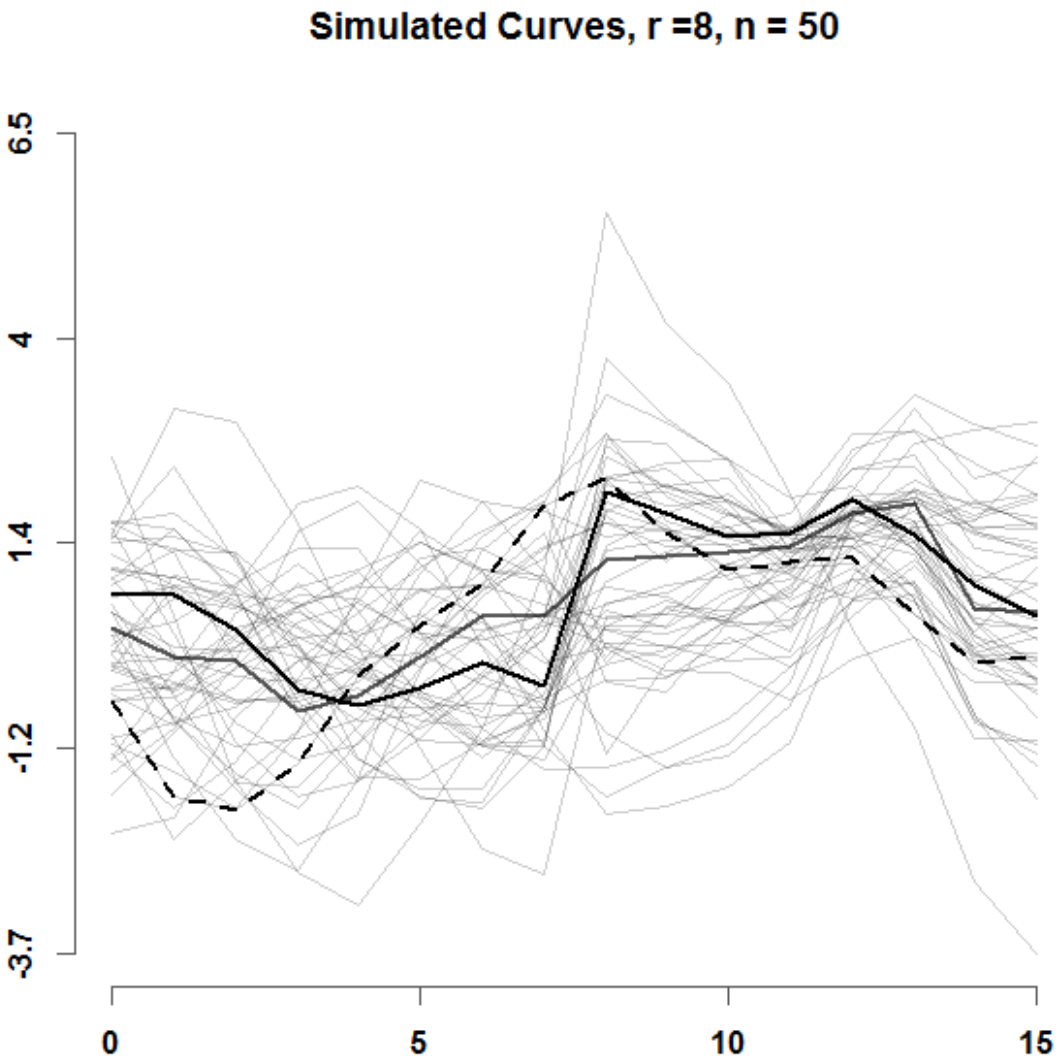

Figure 3. Illustrated are a sample of 50 Simulated Curves with r = 8. Three curves have been highlighted (using a black curve, gray curve, and a dashed curve) for illustrative purposes.

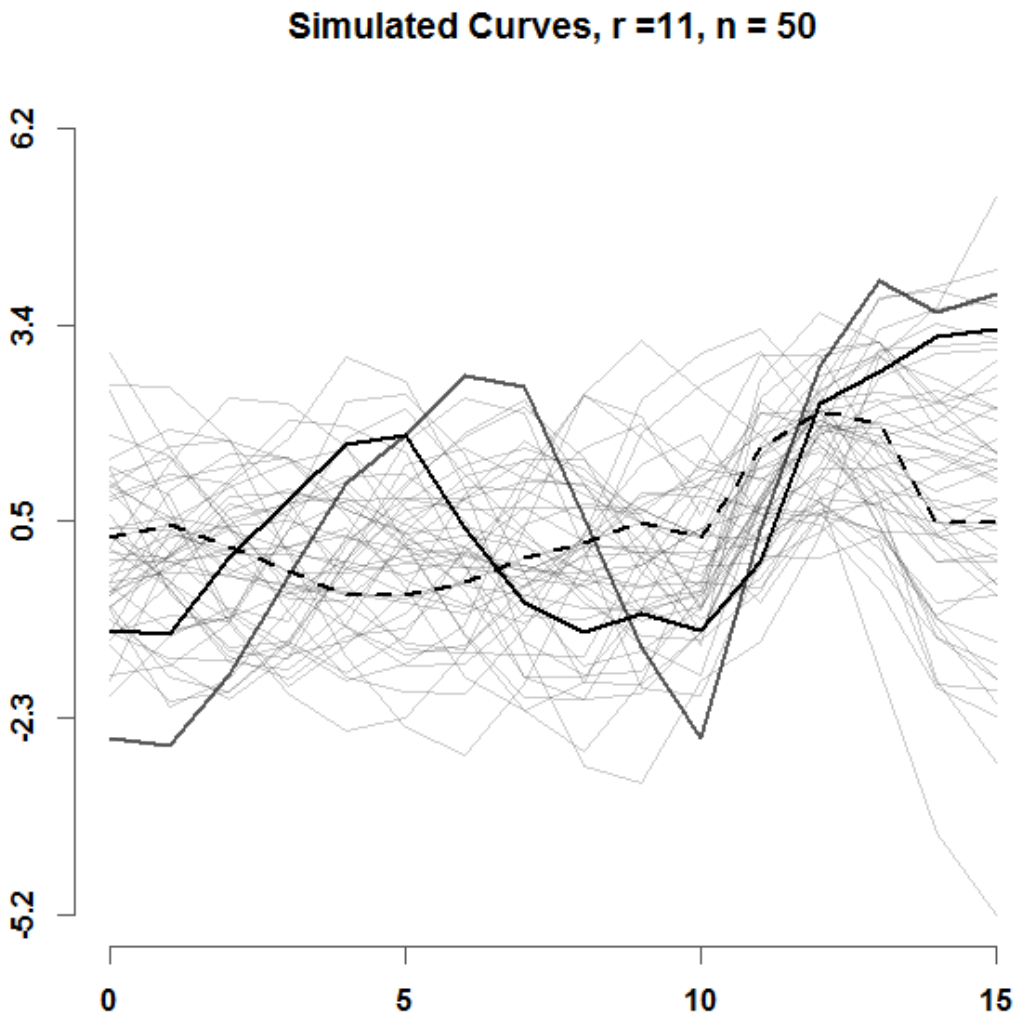

Figure 4. Illustrated are a sample of 50 Simulated Curves with r = 11. Three curves have been highlighted (using a black curve, gray curve, and a dashed curve) for illustrative purposes.

5.2. Construction of Dynamic Prediction Intervals

After generating the dynamic predictions $\hat{Y}_i(\tilde{t})$, we construct approximate $C = 100(1-\alpha)\%$ pointwise prediction intervals of the form

$$\hat{Y}_i(\tilde{t}) \pm t^*_{M-r} * \sqrt{Var\{\hat{Y}_i(\tilde{t})\} + Var\{\epsilon_i(\tilde{t})\}}.$$

The data were analyzed and methods are compared using the metrics defined in the Section 4. The metrics of interest are the Integrated Mean Prediction Error (IMPE), the mean prediction interval width, and the average coverage. The IMPE is the mean of the squared residuals corresponding to all curves, taken over the time points where predictions were actually made. To measure the mean prediction interval width, we average the width of each prediction interval over the n curves and $M - r$ time points. The interval coverage was assessed in numerical simulations when taking 100 simulated datasets, where $Var\{\epsilon_i(\tilde{t})\}$ was known, since it is specified that $\epsilon_i(\tilde{t}) \sim N(0,0.22^2)$.

The following pages contain figures which depict dynamic predictions and prediction intervals (for a specified curve) for all six dynamic prediction methods that were considered, with data simulated using Model (4) and setting A, with n = 25 curves and cutoff point r = 8.

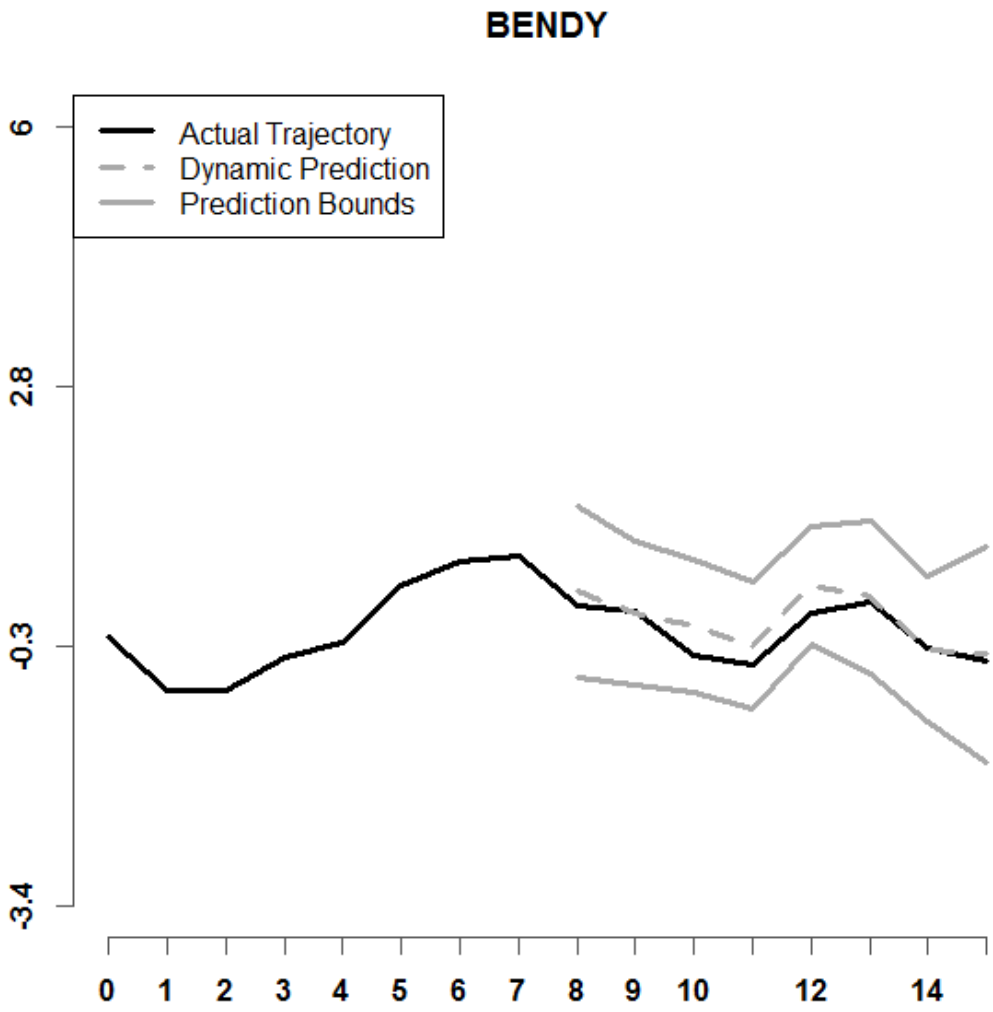

Figure 5. Depicted is a prediction interval for BENDY, with r = 8, n = 25, setting A. Model (4) is used.

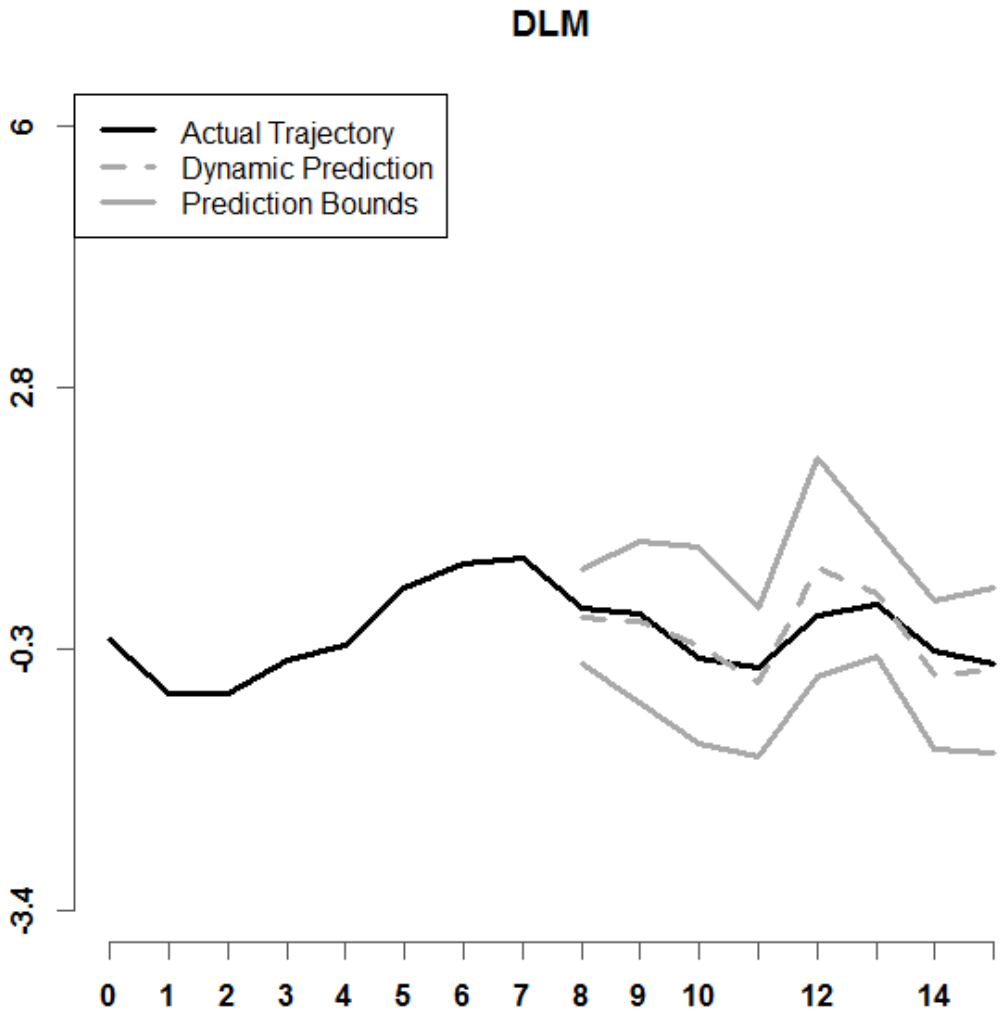

Figure 6. Shown above is a prediction interval for DLM, with r = 8, n = 25, setting A. Model (4) is used.

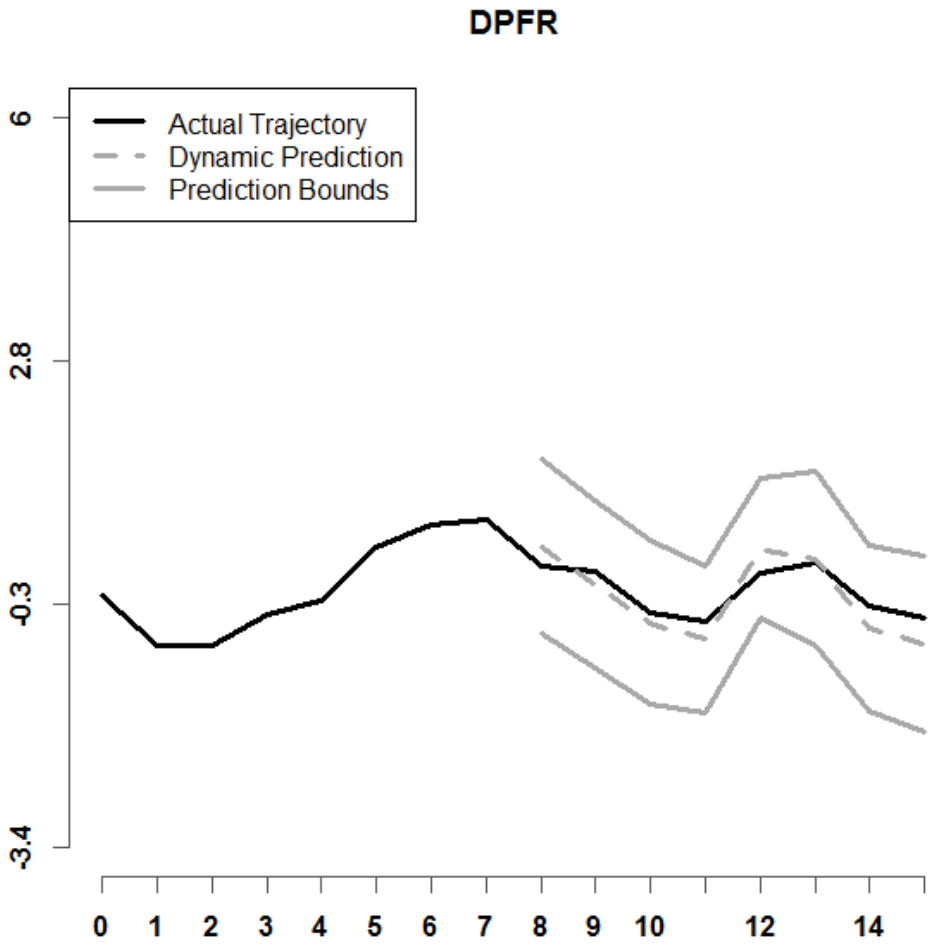

Figure 7. Depicted is a prediction interval for DPFR, with r = 8, n = 25, setting A. Model (4) is used.

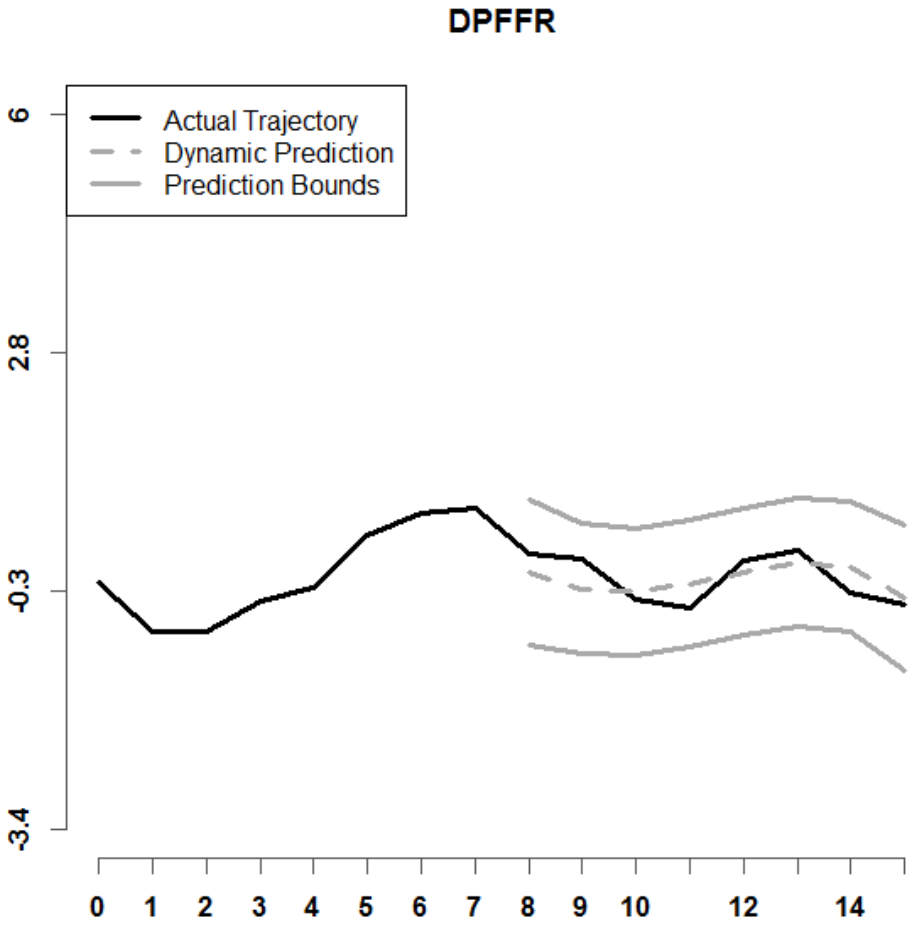

Figure 8. Illustrated is a prediction interval for DPFFR, with r = 8, n = 25, setting A. Model (4) is used.

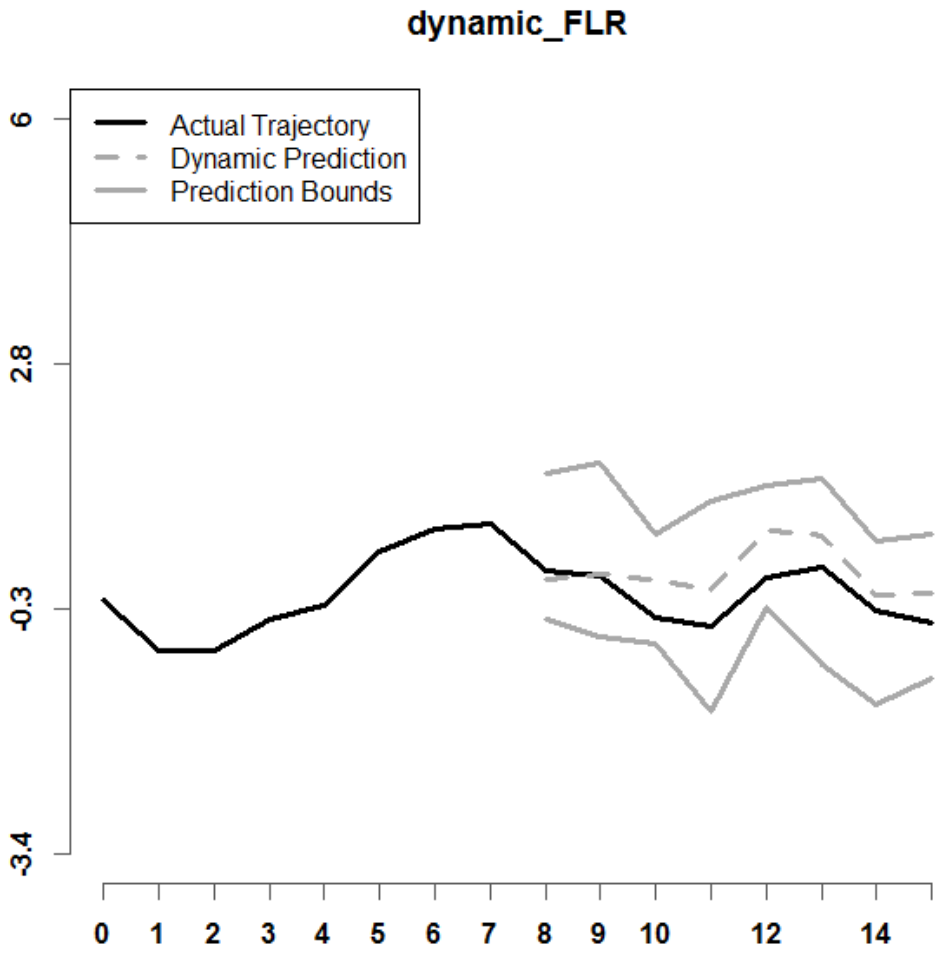

Figure 9. Shown above is a prediction interval for dynamic_FLR, with r = 8, n = 25, and Setting A. Model (4) is used.

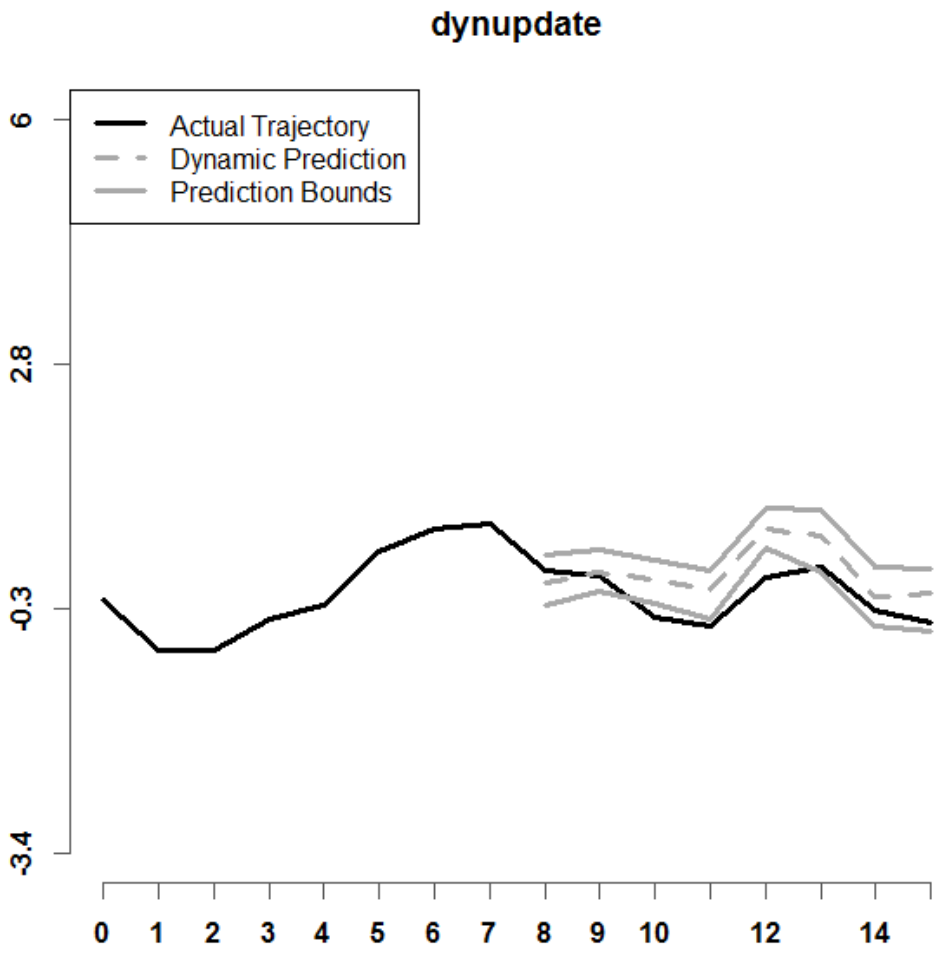

Figure 10. Shown above is a prediction interval for dynupdate, with r = 8, n = 25, and Setting A. Model (4) is used.

Examples of prediction intervals for all six dynamic prediction methods examined in this paper are shown in Figures 5 through 10. Model (4) is used. The lower and upper bounds are depicted as solid dark grey curves. The historical functional data for a specified sample i is shown as a solid black curve. The predicted trajectory is depicted as a dashed dark grey curve. It can be observed that the prediction interval for DPFFR has a stable width that contains the true trajectory of the curve. For this particular curve, it can also be

observed that the true trajectory of the curve was contained within all of the prediction intervals, except for the prediction interval generated by dynupdate.

The next set of figures depict prediction intervals for all six dynamic prediction methods for a simulated dataset using model (1) and setting A, with n = 25 curves and cutoff point r = 8.

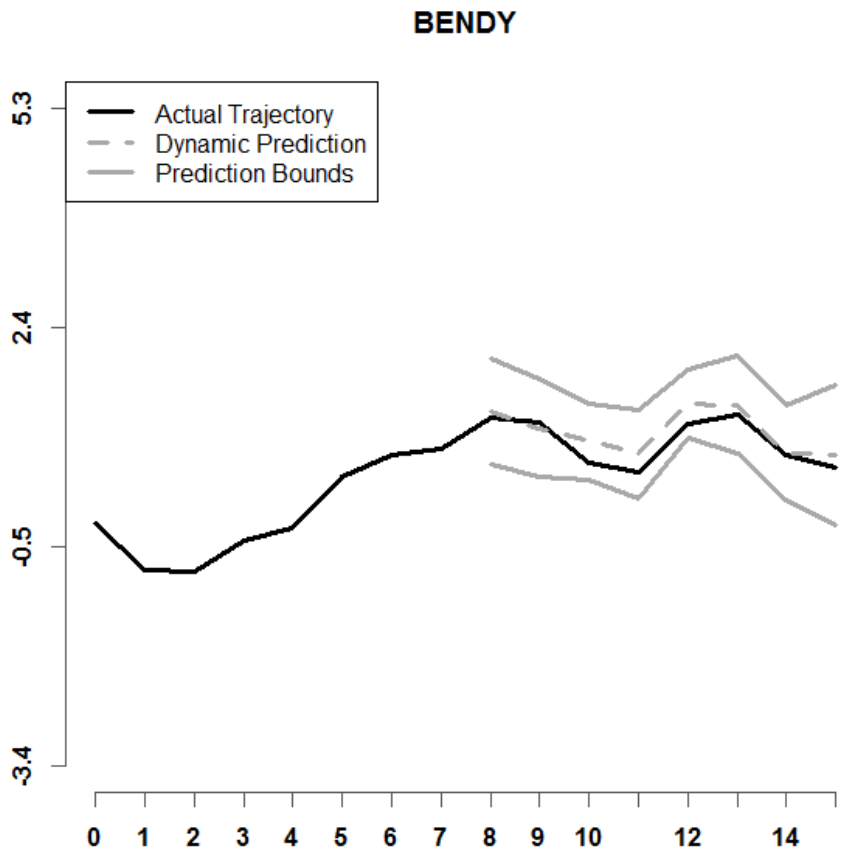

Figure 11. Shown above is a prediction interval for BENDY, with r = 8, n = 25, and Setting A. Model (1) is used.

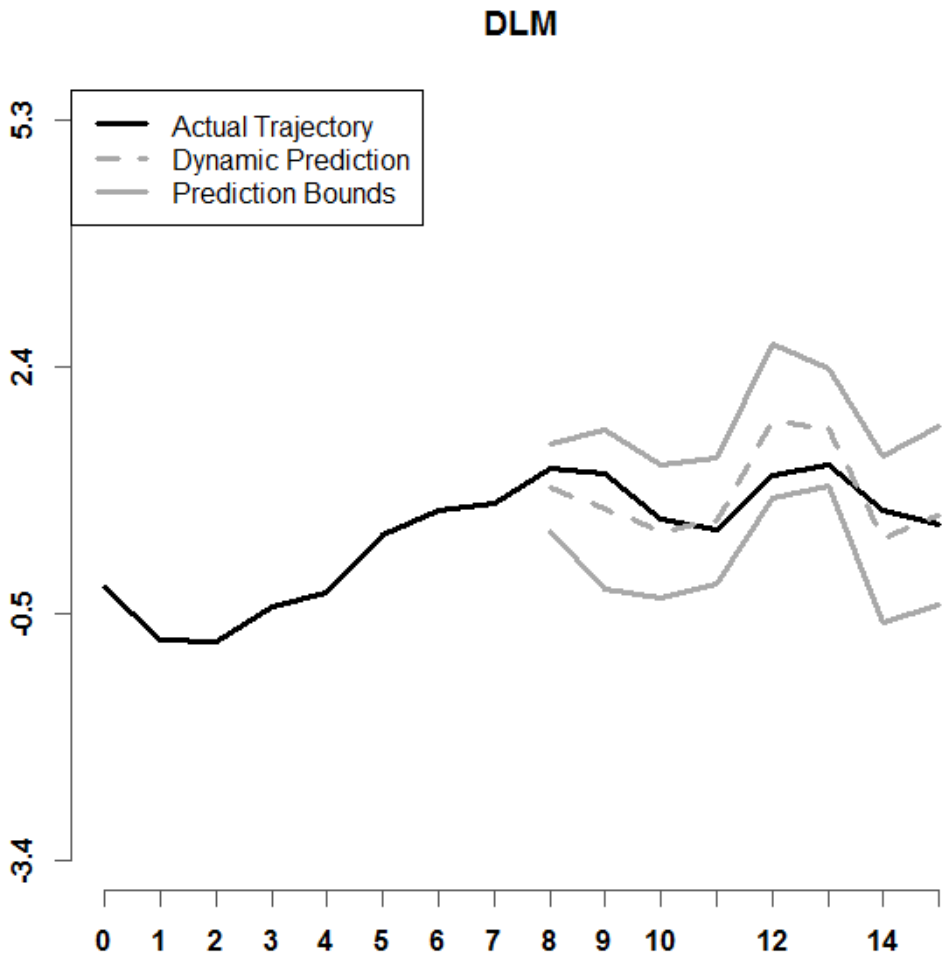

Figure 12. Depicted is a prediction interval for DLM, with r = 8, n = 25, and Setting A. Model (1) is used.

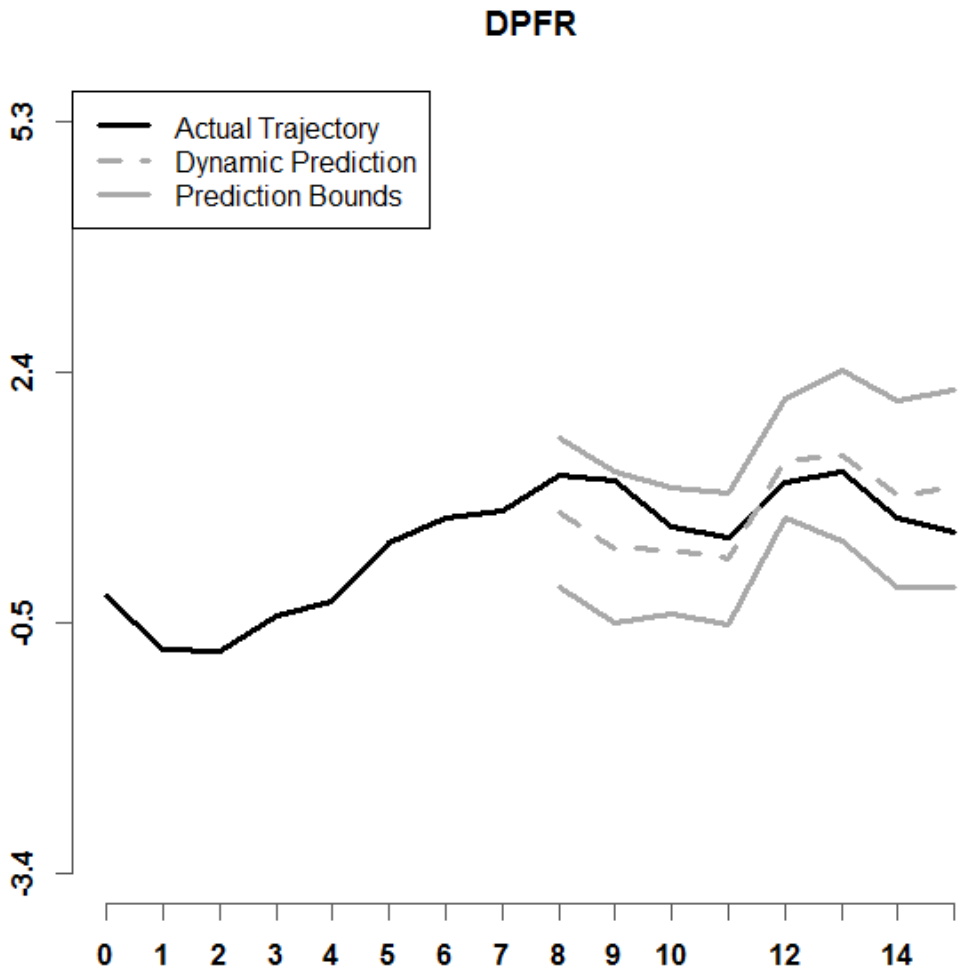

Figure 13. Depicted is a prediction interval for DPFR, with r = 8, n = 25, and Setting A. Model (1) is used.

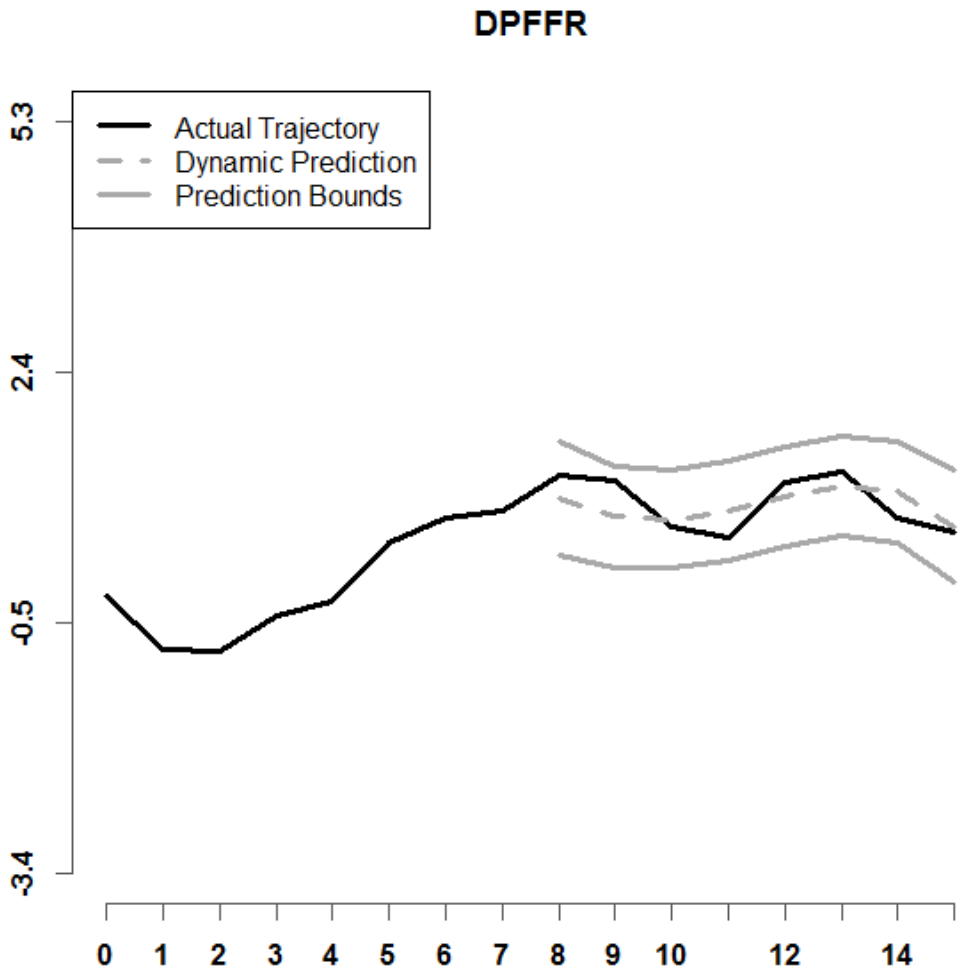

Figure 14. Illustrated is a prediction interval for DPFFR, with r = 8, n = 25, and Setting A. Model (1) is used.

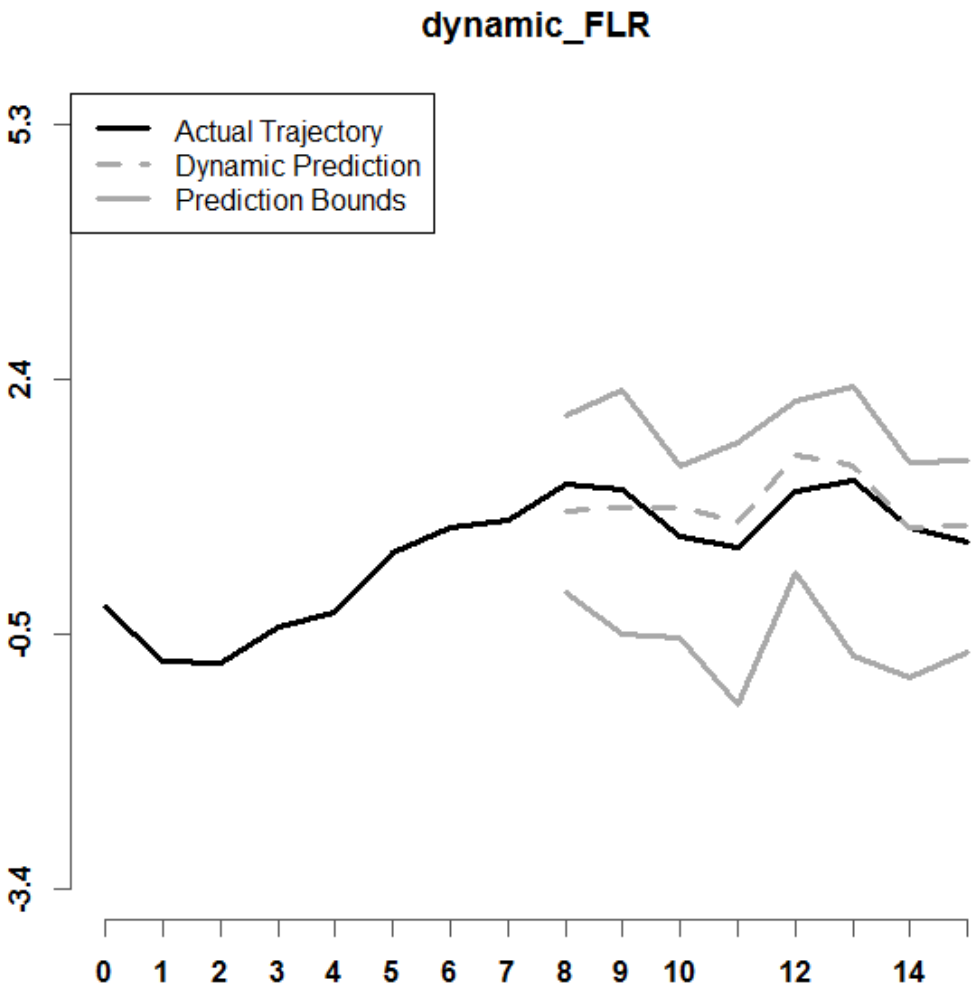

Figure 15. Shown above is a prediction interval for dynamic_FLR, with r = 8, n = 25, and Setting A. Model (1) is used.

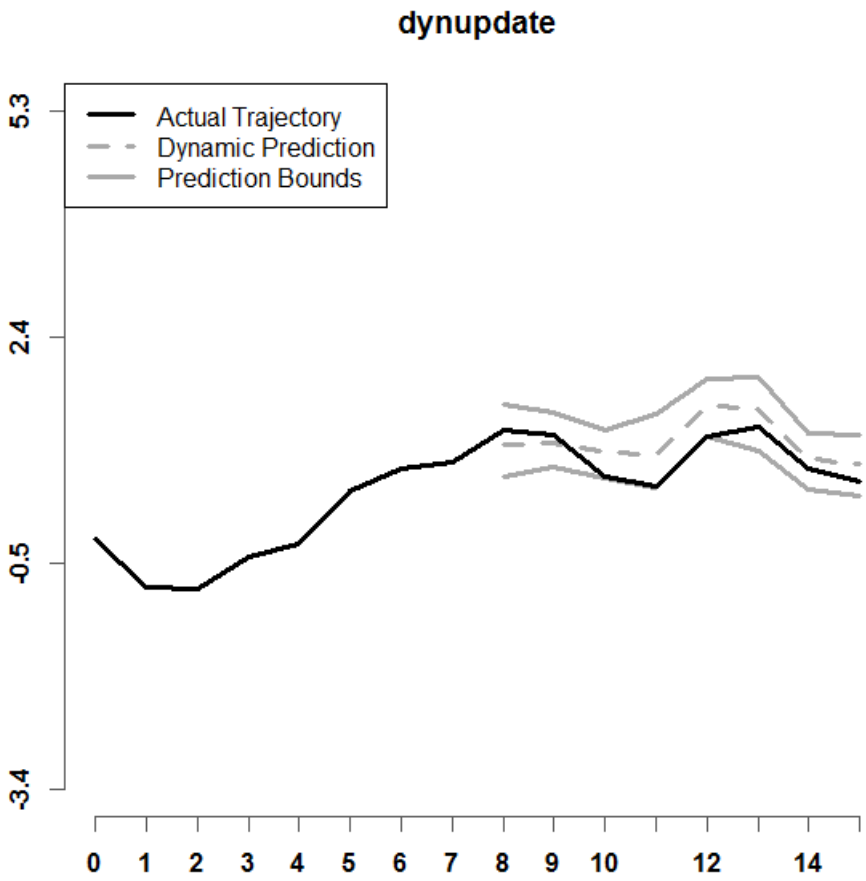

Figure 16. Depicted is a prediction interval for dynupdate, with r = 8, n = 25, and Setting A. Model (1) is used.

Figures 11 through 16 display examples of prediction intervals for all six dynamic prediction methods for one simulated dataset, using model (1), which includes two functional predictors and two scalar covariates. It can be seen in Figure 14 that the width of the DPFFR prediction interval is consistently narrow, and the DPFFR prediction interval contains the true trajectory of the curve. In Figure 16, the lower prediction interval bound for dynupdate partially overlaps with the true trajectory of the curve, which indicates that

in this case, the prediction interval was close to not containing the true trajectory of the curve. The other prediction intervals shown all contained the true trajectory of the curve.

5.3. Simulation Results

For every combination of simulation settings, model (4) was used to generate 100 simulated datasets. The number of curves n, cutoff point r, confidence level $C = 100(1-\alpha)\%$, and data structure setting were changed across the different simulation scenarios. The results for specific cases are summarized in the tables below, which show the metrics of interest for each case. All simulated cases can be viewed in the Appendix.

5.3.1. Increasing the Sample Size

The number of curves n was varied across the different simulation scenarios that were considered in this thesis. The table below displays the metrics for a simulated case where the number of curves was increased from $n = 25$ to $n = 50$.

Table 2. *Simulation Results for increasing the number of curves from n = 25 to n = 50. Results are for setting A, C = 95%, r = 8. Model (4) is used.*

	n = 25				n = 50			
Method	IMPE	AC	AW	CPU	IMPE	AC	AW	CPU
BENDY	0.15	0.95	1.58	0.97	0.14	0.95	1.46	1.77
DLM	0.16	0.95	1.76	1.38	0.07	0.95	1.10	2.53
DPFR	0.19	0.89	1.36	5.72	0.13	0.90	1.19	10.92
DPFFR	0.09	0.94	1.16	7.38	0.09	0.94	1.09	14.08
Dynamic_FLR	0.17	0.97	2.28	12.24	0.14	0.96	1.87	29.79
dynupdate	0.10	0.69	0.63	21.04	0.12	0.70	0.70	70.10

It can be observed in Table 2 that, for this case, as the number of curves increased, the IMPE either decreased or remained the same for all dynamic prediction methods except dynupdate. Similarly, as the number of curves increased, the AW of the prediction intervals decreased for most of the dynamic prediction methods. Additionally, as the number of curves increased, the CPU time for each dynamic prediction method increased, which makes sense, as increasing the number of curves would cause the methods to take longer to run. Similar results were noticed when examining other simulated cases; in the other cases observed in this thesis, when the number of curves increased, the IMPE, AW, and CPU responded as described here. Other cases may be viewed in the Appendix (Table A1 for IMPE, Tables A2 – A4 for some other cases).

5.3.2. Changing the Data Structure

The data structure was also varied across the different simulation scenarios A and B.

Table 3. *Simulation Results for changing the data structure from Setting A to Setting B. Results are for n=25, C = 95%, r = 8. Model (4) is used.*

	Setting A				Setting B			
Method	IMPE	AC	AW	CPU	IMPE	AC	AW	CPU
BENDY	0.15	0.95	1.58	0.97	0.59	0.95	3.16	0.88
DLM	0.16	0.95	1.76	1.38	0.16	0.95	1.76	1.25
DPFR	0.19	0.89	1.36	5.72	0.37	0.82	1.57	5.05
DPFFR	0.09	0.94	1.16	7.38	0.09	0.94	1.15	5.71
Dynamic_FLR	0.17	0.97	2.28	12.24	2.34	0.95	7.15	8.97
dynupdate	0.10	0.69	0.63	21.04	1.61	0.84	3.70	12.52

Table 3 shows that for this simulated case, when the data structure is changed from Setting A to Setting B, the IMPE increases for all methods except DPFFR, which has the same IMPE in both data structures. When the data structure is changed from Setting A to Setting B, the AW also increases for most methods, except for DPFFR. The AC remains at a similar level for BENDY, DLM, DPFFR, and dynamic_FLR for both Setting A and Setting B. For n = 25 curves, changing the setting did not greatly impact the CPU time. These trends were noticed in the simulated cases that were checked in this thesis, and more detailed tables on these simulated cases can be found in the Appendix (Table A1 for IMPE, Tables A2 – A4 for AC, AW, and CPU for other cases).

5.3.3. Increasing the Confidence Level

The confidence level was varied throughout the different simulation scenarios that were considered. It was of interest to see if any of the metrics changed when the confidence level C increased.

Table 4. *Simulation Results for changing the confidence level from C=90% to C=95%. Results are for n=25, Setting A, r = 11. Model (4) is used.*

	C = 90%		C = 95%	
Method	AC	AW	AC	AW
BENDY	0.90	2.35	0.94	2.84
DLM	0.90	2.94	0.95	3.97
DPFR	0.84	1.15	0.92	1.47
DPFFR	0.95	1.00	0.99	1.27
Dynamic_FLR	0.92	2.81	0.96	3.35
dynupdate	0.56	0.79	0.63	0.93

Table 4 displays the AC and AW from a simulated case with n = 25 curves, Setting A, and r = 11. It can be seen that in this simulated case, when the nominal coverage level increased, the AW increased for all methods. Since the AW of the prediction intervals increased, there was a corresponding increase in the AC, as wider intervals would have greater coverage. Changes in the nominal coverage level did not have any impact on the IMPE. These trends are also observable in all of the other cases that were simulated for the purpose of this thesis, and more detailed tables on these simulated cases for model (4) can be found in Tables A2-A4 in the Appendix.

5.3.4. Changing the Cutoff Point, R

Two values for the cutoff point were considered for simulations. Several cases were simulated with r = 8, and the remaining cases were simulated with r = 11. It was of interest to see how the metrics changed when the cutoff point r increased.

Table 5. *Simulation Results for increasing the cutoff point from r = 8 to r = 11. Results are for n=50, C = 95%, Setting B. Model (4) is used.*

	r = 8			r = 11		
Method	IMPE	AC	AW	IMPE	AC	AW
BENDY	0.55	0.95	2.95	2.38	0.95	6.13
DLM	0.07	0.95	1.10	0.09	0.95	1.20
DPFR	0.25	0.82	1.31	0.49	0.79	1.58
DPFFR	0.09	0.94	1.08	0.06	0.99	1.21
Dynamic_FLR	2.01	0.95	6.53	6.58	0.95	11.76
dynupdate	1.71	0.86	4.12	5.82	0.92	8.87

Table 5 displays the IMPE, AC, and AW from a simulated case with n = 50 curves, Setting B, and C = 95%. In this simulated case, when the cutoff point increased from r = 8 to r = 11, the IMPE increased for all methods, except DPFFR. The AC for DPFFR and dynupdate increased when r increased for this case. It should also be noted that the AW of the prediction intervals increased when the cutoff point increased from r = 8 to r = 11. This case is an example of a pattern that was noticed throughout the simulated cases. Throughout the cases that were simulated, when the cutoff point was increased from r = 8 to r = 11, the AW tended to increase, and the AC for DPFFR tended to increase as well. For the detailed metrics on the model (4) simulated cases, see Tables A1-A4 in the Appendix.

5.3.5. Adding Scalar Covariates

Simulations were also conducted for model (1), which includes the two scalar covariates $W_{i1} = I\{U[0,1] \geq 0.75\}$ and $W_{i2} \sim N(0, 0.1^2)$ in addition to the two functional predictors $Y_i(t)$ and $Z_i(t)$. Model (1) is given by:

$$Y_i(\tilde{t}) = W_{i1}\gamma_1 + W_{i2}\gamma_2 + \zeta(\tilde{t}) + \int_{\mathrm{T}} Y_i(t)\beta(\tilde{\mathrm{t}}, \mathrm{t})\, dt + \int_{\mathrm{T}} Z_i(t)\delta(\tilde{\mathrm{t}}, \mathrm{t})dt + \epsilon_i(\tilde{\mathrm{t}}). \quad (1)$$

It was of interest to see how the metrics changed when the scalar covariates were added to the model. The table below compares metrics between model (4), which has two functional covariates and no scalar covariates, and model (1), which has two functional covariates and two scalar covariates.

Table 6. *Simulation Results for adding scalar covariates to the model. Results are for n=25, r=11, C = 95%, Setting A.*

	Model (4)			Model (1)		
Method	IMPE	AC	AW	IMPE	AC	AW
BENDY	0.52	0.94	2.84	0.57	0.95	3.00
DLM	0.55	0.95	3.97	3.93	0.95	26.66
DPFR	0.19	0.92	1.47	0.52	0.81	1.89
DPFFR	0.07	0.99	1.27	0.07	0.99	1.30
Dynamic_FLR	0.48	0.96	3.35	0.74	0.95	4.02
dynupdate	0.27	0.63	0.93	0.43	0.64	1.22

When the scalar covariates were considered, there were some changes in the metrics reported. In this simulated case, adding the scalar covariates to the simulation model resulted in an increase in the IMPE for all methods except DPFFR, as Table 6 indicates. When the scalar covariates were added to the model, the AW increased as well; in general, the increase widths can be observed in the tables in the Appendix. The increase in AW for DPFFR was relatively small, while other methods showed a more substantial increase. Adding the scalar covariates did not have any significant impact on the AC for most of the methods. The exceptions to this are DPFR and dynupdate. For all cases simulated in this thesis, the coverage for DPFR decreased when the scalar covariates were added.

Adding the covariates to the model did not have a large impact on the patterns that were observed when changing the sample size, data structure, confidence level, and cutoff point. When n increased from 25 to 50, the IMPE decreased for all methods, and the AW

decreased for all methods except dynupdate. When the data structure was changed from setting A to setting B, the AW increased for BENDY, dynamic_FLR, and dynupdate, while it remained relatively similar for the other methods. Going from setting A to setting B increased the AC of dynupdate, but had little impact on the AC for the other methods. When the confidence level increased from 90% to 95%, the AC and AW of the prediction intervals increased as expected. When r increased from 8 to 11, the IMPE increased for all methods except DPFFR, which decreased. Overall, DPFFR had very low IMPE in the model (1) and model (4) cases, it maintained consistent AW, and it had AC that was close to the desired confidence level. For almost all simulated cases, DPFFR had a CPU time that was lower than those of dynamic_FLR and dynupdate, yet slightly larger than the CPU time for DPFR. Among the dynamic prediction methods that treat the predicted response as a function (DPFFR, dynamic_FLR, and dynupdate), DPFFR performed comparatively well in these simulated cases and was preferred. For detailed metrics on the model (1) simulated cases, see Tables A5-A8 in the Appendix.

6. Application to Financial Stock Data

The dynamic prediction methods were used to analyze data on monthly stock highs from two well-known companies: Microsoft (abbreviated as MSFT) and IBM. The data from Microsoft (Microsoft Historical Prices, 2016) and IBM (IBM Historical Prices, 2016) were obtained from an online database provided by Yahoo Finance. The monthly high stock price is the maximum value that the stock was known to have for a given month. From 1987 to 2015, the monthly high stock price for IBM and MSFT for each month was recorded, resulting in 12 data points per year for 29 years. Therefore, for this application, the sample size is n = 29, and the number of points M for the time domain is 12. This type

of data is known as time series functional data (Shang and Hyndman, 2016). At the outset of this application, the response $Y_i(t)$ is taken to be the MSFT stock high during month t of year i. Note that for the DPFFR method, the IBM monthly stock highs were taken to be the functional covariates $Z_i(t)$.

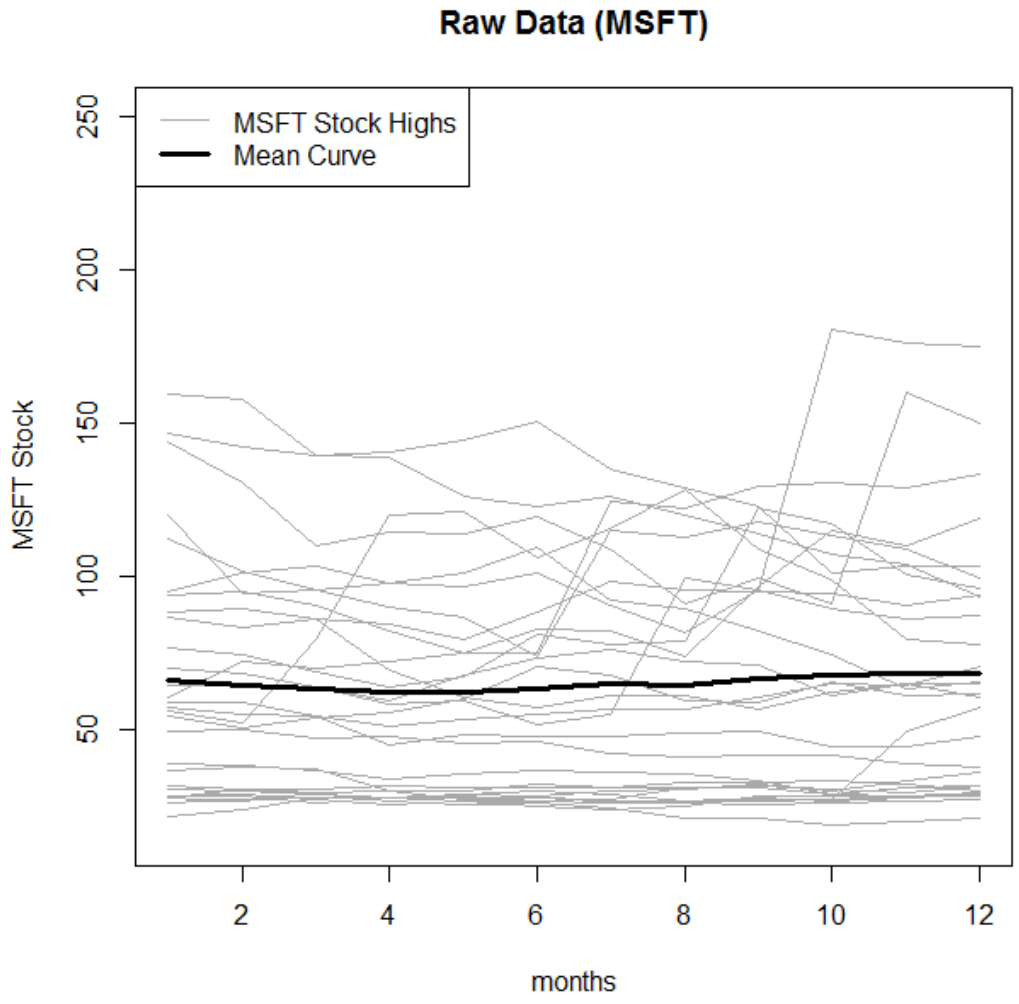

Figure 17. Shown is the raw data for MSFT Stock Highs over n = 29 years for each month.

Figure 17 displays the raw data on Microsoft stock highs. The black curve is the mean curve (Ramsay and Silverman, 2005; Ramsay et al., 2017), which displays the average MSFT stock high for each month. The mean curve remains relatively constant as the

months progress, while individual curves display variability with respect to the mean functional curve. Figure 18 displays the same information for the IBM stock highs. The mean function for the IBM stock highs shows a slight increase after month 7.

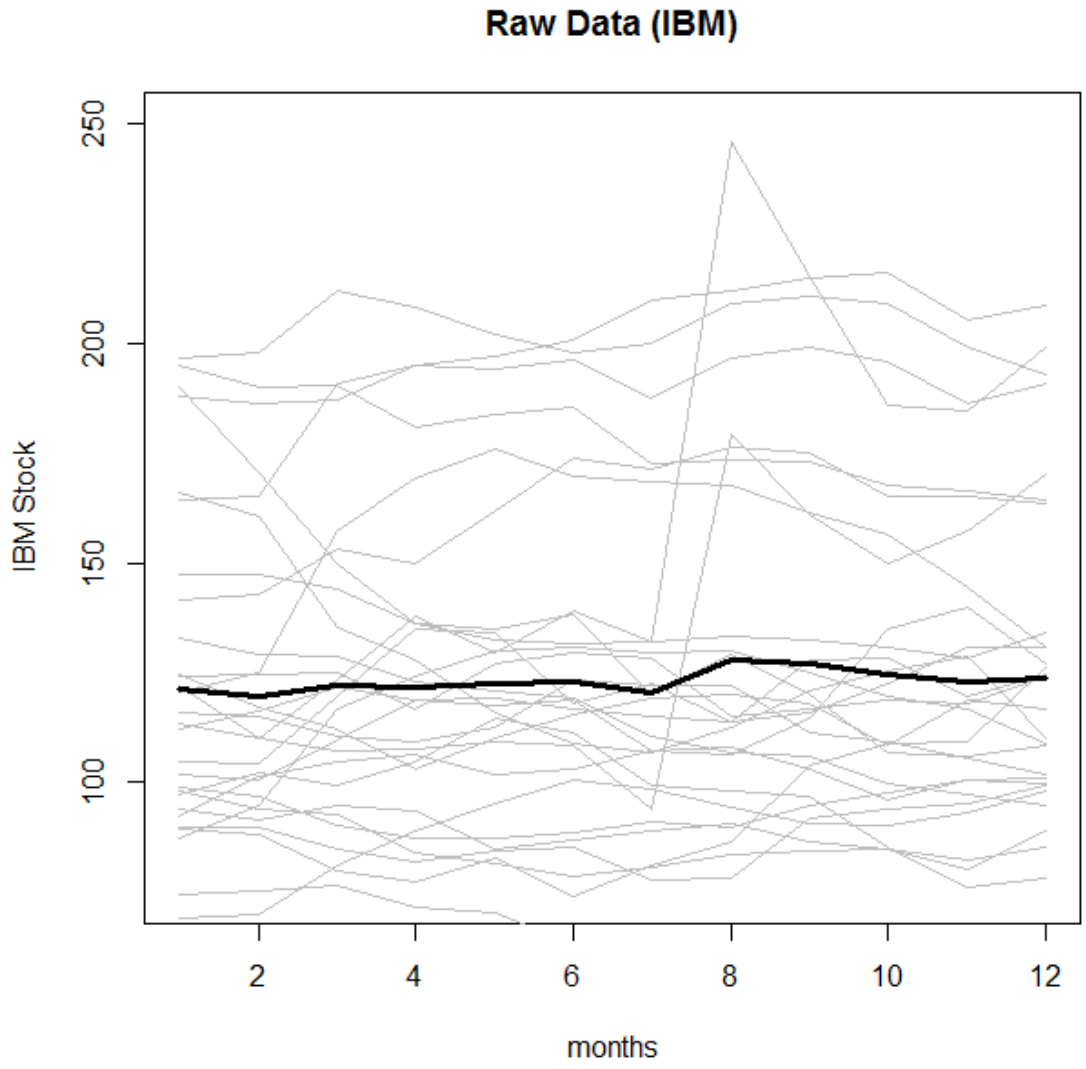

Figure 18. Shown is the raw data for IBM Stock Highs over n = 29 years for each month.

6.1. Functional Principal Components Analysis

The variance of the IBM and MSFT data was analyzed using Functional Principal Components Analysis (FPCA). FPCA can be used to obtain estimates of the functional data curves, denoted as $\hat{Y}_i(t)$. Once the estimates are known, the error terms can be estimated as $\hat{\epsilon}_i(t) = \hat{Y}_i(t) - Y_i(t)$, and then the variance of the error terms can be estimated. FPCA was used in this study to obtain an estimate of $Var\{\epsilon_i(\tilde{t})\}$, so that prediction intervals could be obtained. This method considers the covariance V at two time points t and s, and states that the covariance can be decomposed in the following manner:

$$V(t,s) = \sum_{k=1}^{K} d_k \xi_k(t) \xi_k(s),$$

where the d_k are eigenvalues and the ξ_k are eigenfunctions. Here, K is the number of functional principal components that the covariance is decomposed into. Each d_k gives the amount of variation in the direction of $\xi_k(t)$, and $\frac{d_k}{\sum d_k}$ yields the proportion of variance explained by the k^{th} functional principal component.

Then, by the Karhunen-Loève Theorem, the functional samples can be reconstructed as:

$$Y_i(t) = \mu(t) + \sum_{k=1}^{\infty} f_k \, \xi_k(t),$$

where $\mu(t)$ is the mean function, and the coefficients f_k and eigenfunctions $\xi_k(t)$ are obtained from the covariance decomposition in FPCA. In practice, it is sufficient to select the first K principal components, if most of the variance is explained by them, since a finite number of principal components is needed for estimation of the curves. Once K is selected, the curves $Y_i(t)$ may be estimated as:

$$\hat{Y}_i(t) = \hat{\mu}(t) + \sum_{k=1}^{K} f_k \hat{\xi}_k(t).$$

After the FPCA breakdown for variance was performed, it was revealed that the first functional principal component explained 97.7% of the variance, the second component explained 1.5% of the variance, and the third component explained 0.6% of the variance (Ramsay and Silverman, 2005; Ramsay et al., 2017; Goldsmith et al., 2017). Together, the first three components explain 99.8% of the variance. Since the first functional principal component explains so much of the variance, it would be sufficient to include only the first principal component or only the first two principal components, but for completeness, the first three principal components were provided.

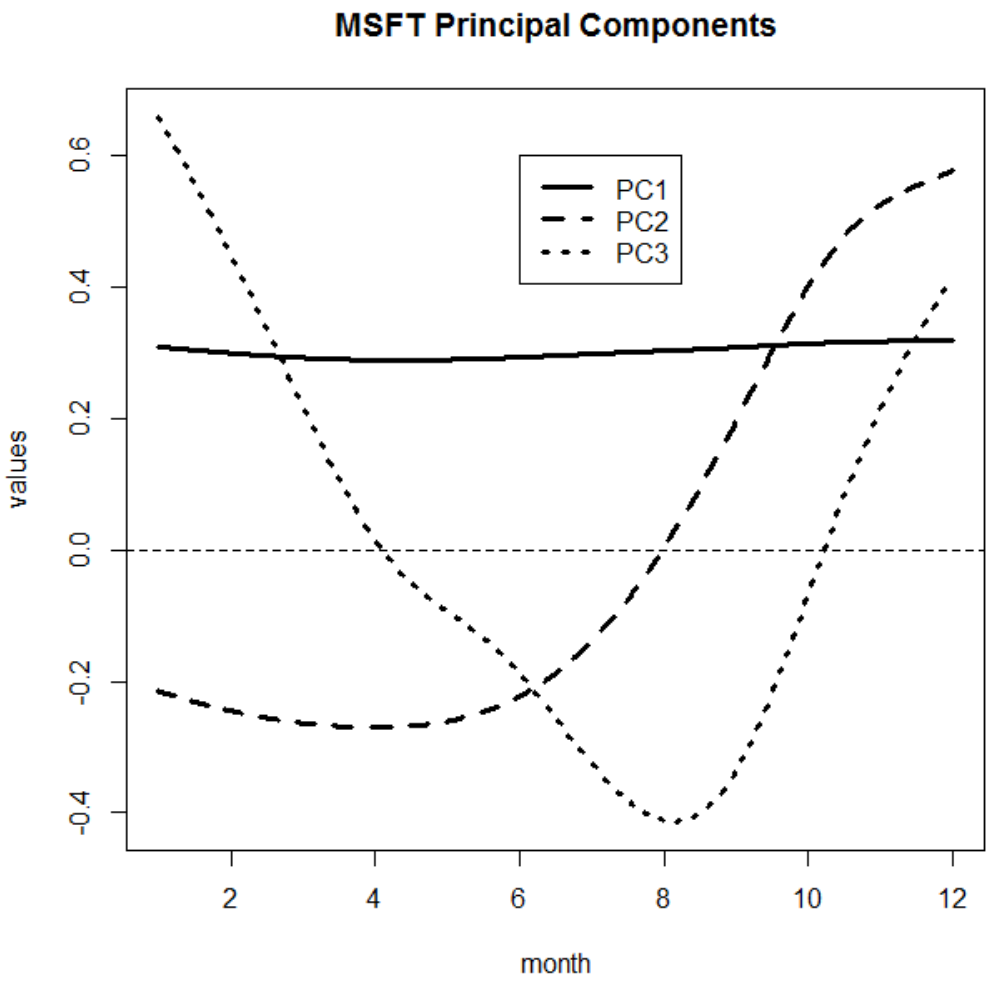

Figure 19. Illustrated are the first three principal components for Microsoft (MSFT) data (PC1, PC2, and PC3, respectively).

PC1 was a main vertical shift depicted in the above figure. It represents a relatively constant vertical shift in the MSFT stock highs from year to year. PC2 contrasts the beginning of the year with the end of the year. PC3 represents a minor (0.6% of variability explained) seasonal effect on the monthly stock highs.

6.2. Results in Data Analysis

Methods DPFFR, dynamic_FLR (Shang, 2015; Shang and Hyndman, 2016), and dynupdate (Shang, 2015; Shang and Hyndman, 2016) were used on this data to predict the monthly stock highs for Microsoft after two cutoff points (r = {4,5}) in this application. These three dynamic prediction methods were selected because it was of interest to treat the predicted monthly stock highs for Microsoft as functions. The results were analyzed using the metrics outlined in the Methods section. The results are summarized in Table 7.

Table 7. *IMPE for three dynamic prediction methods, with r = {4,5}.*

Method	**IMPE**
DPFFR	324.256 (r = 4) 379.758 (r = 5)
dynamic_FLR	436.981 (r = 4) 517.657 (r = 5)
dynupdate	797.374 (r = 4) 814.085 (r = 5)

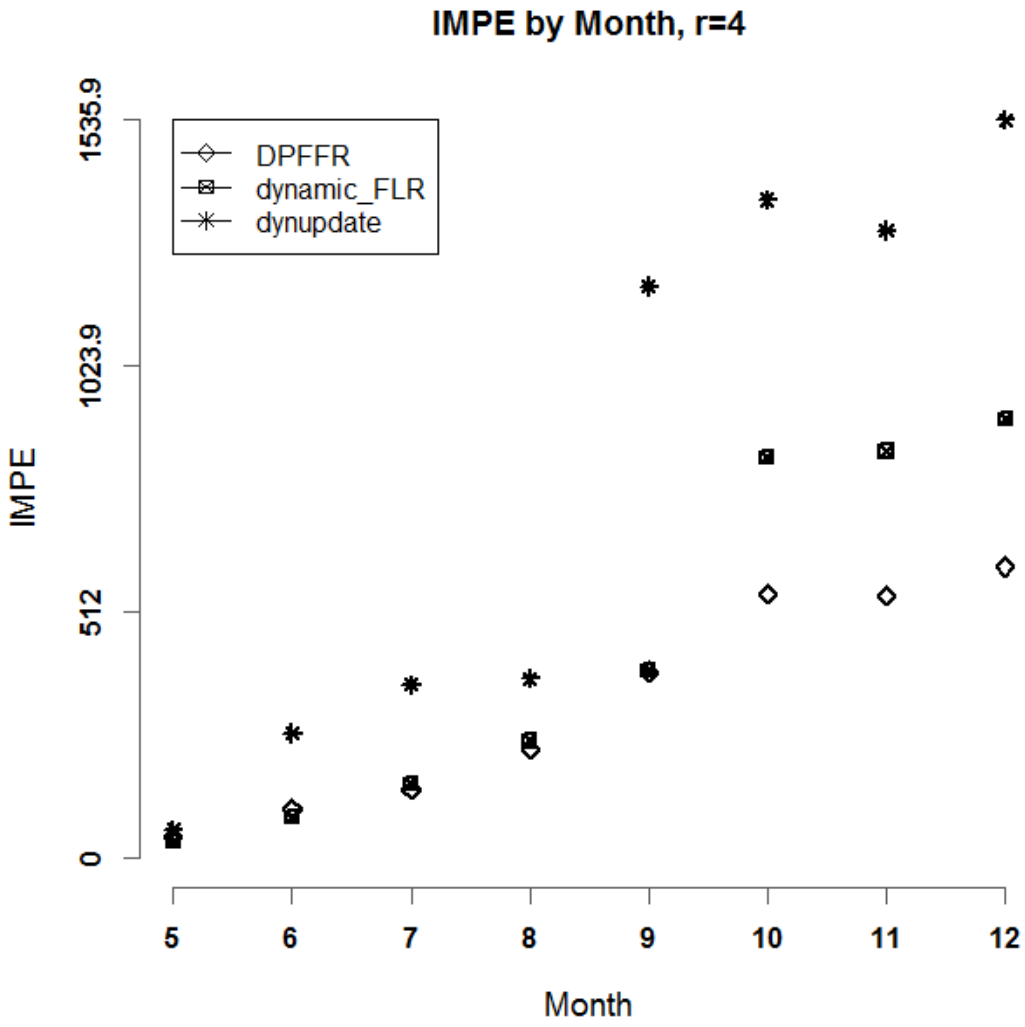

Figure 20. Depicted is a graph of the IMPE, by month, when the cutoff point r = 4.

Results of the application indicate that the IMPE was lower for DPFFR than the other two methods (Figure 20). Additionally, the IMPE seems to increase for months towards the end of the year. One can observe that for all months after month 5 (May), the IMPE of DPFFR is less than or equal to the IMPE for the other two methods (Figure 21). The IMPE is smaller at months close to the cutoff point r.

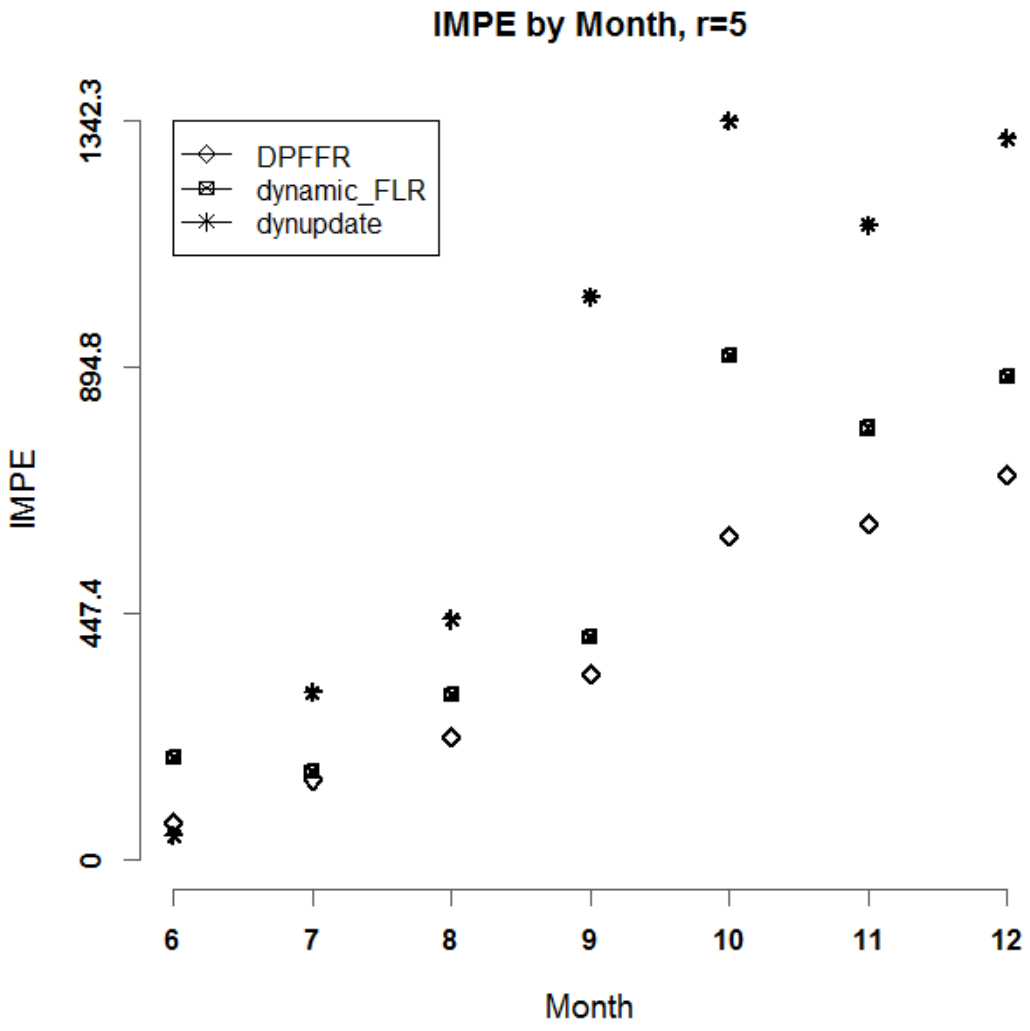

Figure 21. Depicted is a graph of the IMPE, by month, when the cutoff point r = 5.

The next two figures depict dynamic predictions for the MSFT stock highs in a specific year using DPFFR, for cutoff points r = 4 and r = 5. In Figure 22 and Figure 23, we display some examples of dynamic predictions where the predicted trajectories of the MSFT stock highs according to DPFFR are indicated by dashed lines. DPFFR dynamic predictions were close to the actual trajectories.

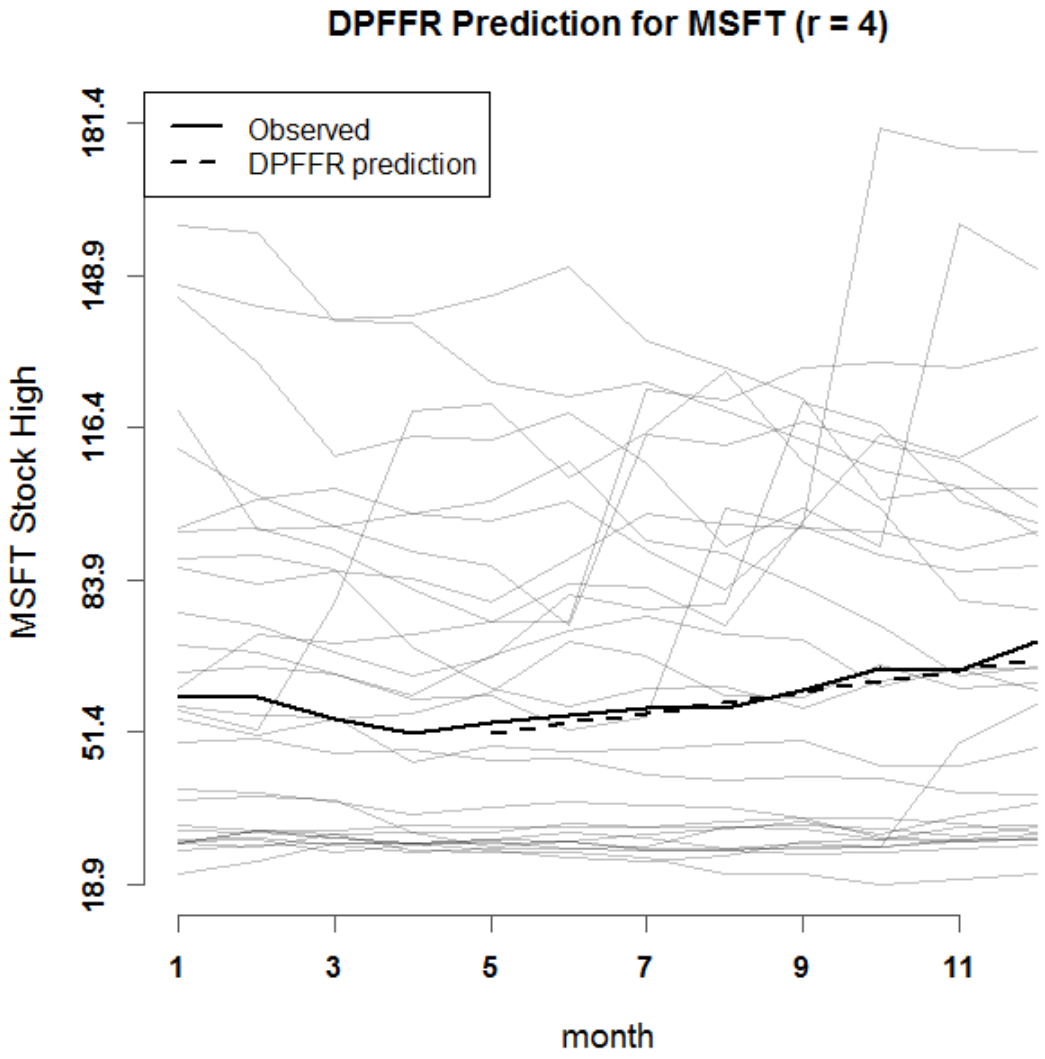

Figure 22. Depicted is a graph of a DPFFR prediction of the MSFT stock high for a specific year, when r = 4.

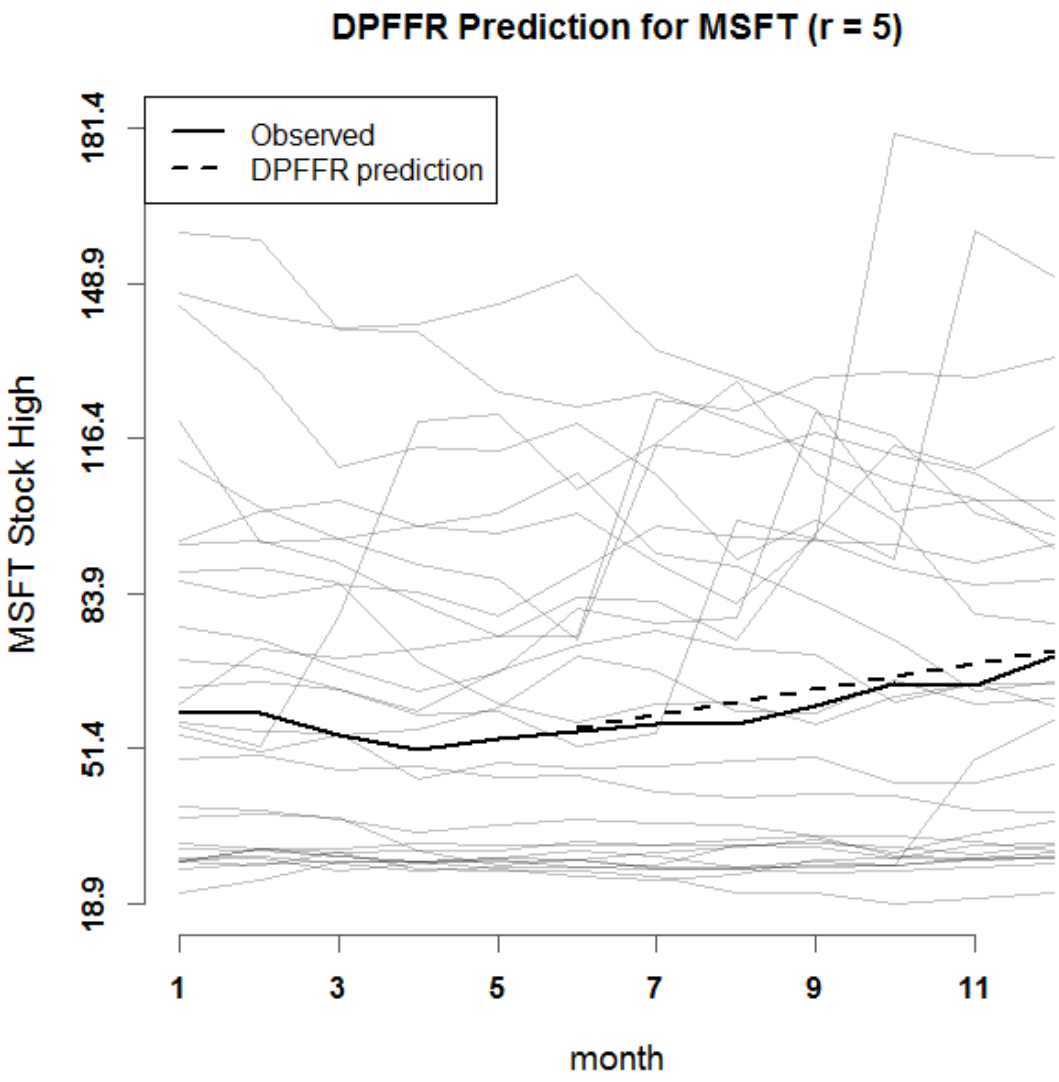

Figure 23. Depicted is a graph of a DPFFR prediction, for a different curve, of the MSFT stock highs when r = 5.

The Intramonthly Stock Returns are a measure of stock performance (Shang, 2015) and were considered for analysis. For this phase of the analysis, the IBM stock highs from 1987 to 2015 were predicted (with the MSFT stocks as predictive information for DPFFR). To perform this kind of analysis, the following transformation (Shang, 2015) is applied to the data:

$$R_i(t) = 100 * \left(\ln\left(Y_i(t)\right) - \ln\left(Y_i(1)\right)\right).$$

$R_i(t)$ is the return for the IBM stock in year i and month t. The return can be thought of as the increase in the value of a stock relative to its price during the first month of the year. Here, $Y_i(t)$ is the IBM stock high during month t of year i. $Y_i(1)$ is the IBM stock high during the first month of year i. Note that it only makes sense to consider $R_i(t)$ over the domain $t \in \{2,3, \dots, M\}$, as $R_i(1) = 0$.

Table 8. *Average Coverage (AC), Average Width (AW), and CPU Time for three dynamic prediction methods, C = 95% confidence, n = 29 curves, with cutoff point r = 7.*

Method	**AC**	**AW**	**CPU Time**
DPFFR	0.897	46.446	10.40
dynamic_FLR	0.940	54.198	11.90
Dynupdate	0.897	41.780	25.31

DPFFR has a faster run time than dynamic_FLR and dynupdate. As far as coverage is concerned, DPFFR performs just as well as dynupdate, and it is relatively similar to dynamic_FLR. The prediction intervals for dynamic_FLR had the largest width out of the three methods. DPFFR had an actual coverage relatively close to the nominal level of 95% for this data.

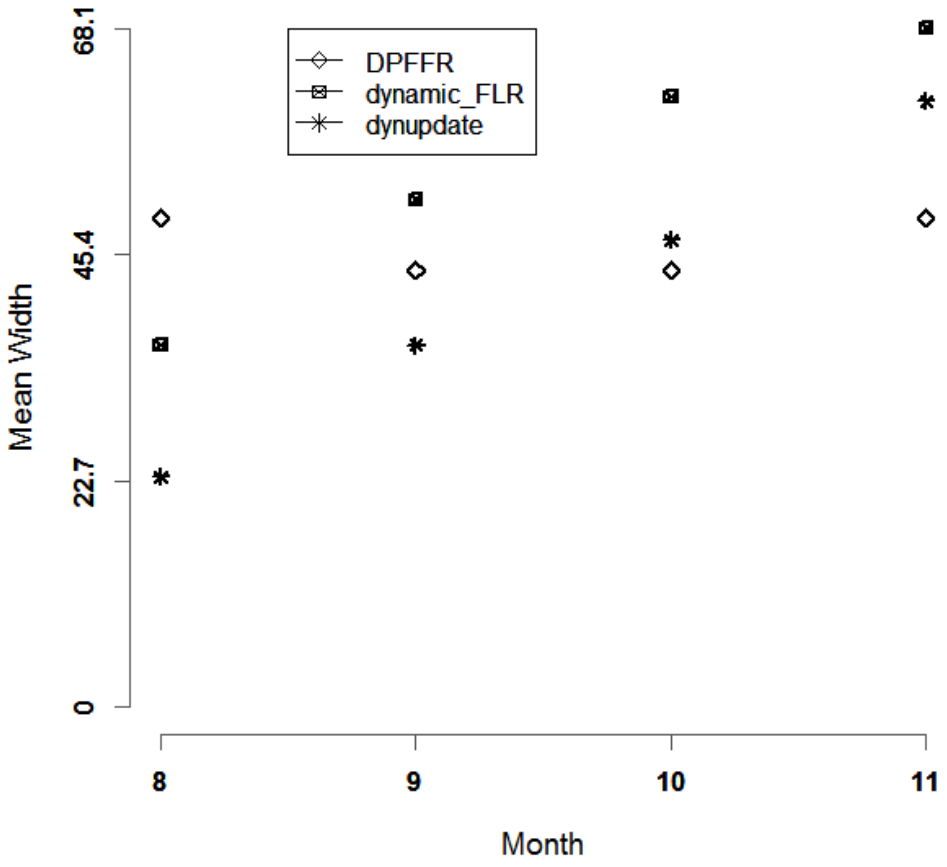

Figure 24. Depicted is a graph of mean prediction interval width for each of the three methods, with cutoff point r = 7 (August).

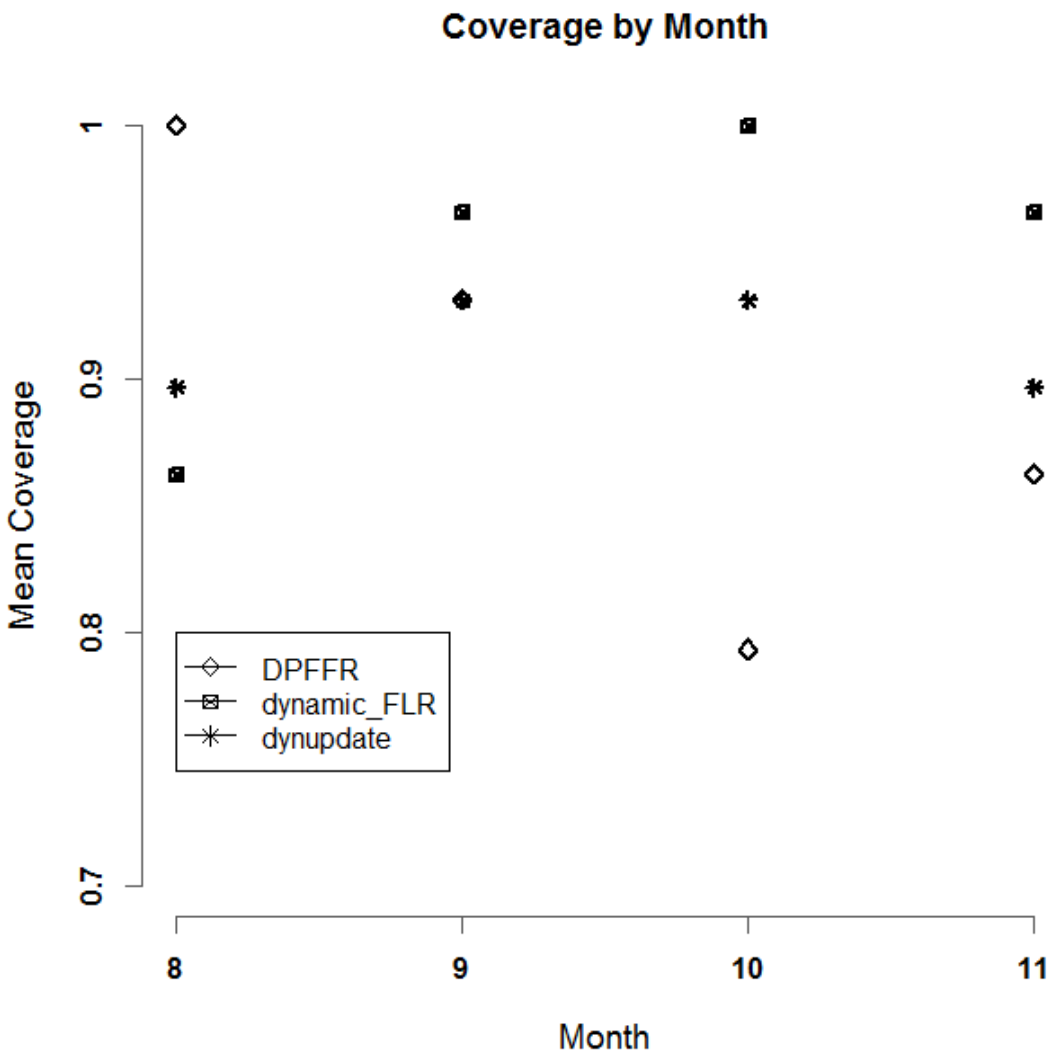

Figure 25. Depicted is a graph of prediction interval coverage for three dynamic prediction methods, for cutoff point r = 7 (August).

The width of DPFFR is relatively constant throughout the months. In September DPFFR had coverage that was greater than the coverage of the other methods, while in November, DPFFR had the lowest coverage. In October, DPFFR and dynupdate have the same mean coverage (the icons overlap). The methods all have mean coverages at or above 0.85 during September, October, and December.

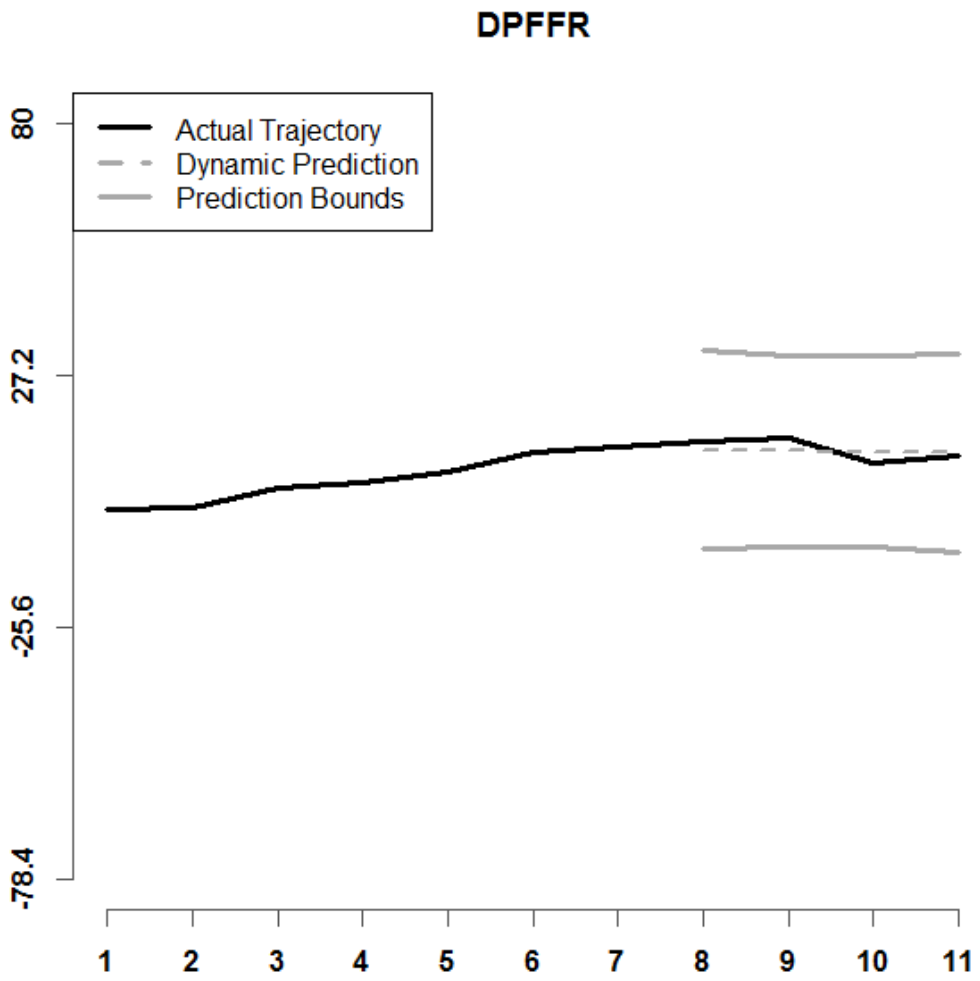

Figure 26. Shown above is a graph of a DPFFR prediction interval for the Intramonthly Stock Returns for IBM for a given year, when r = 7 and C = 95% confidence.

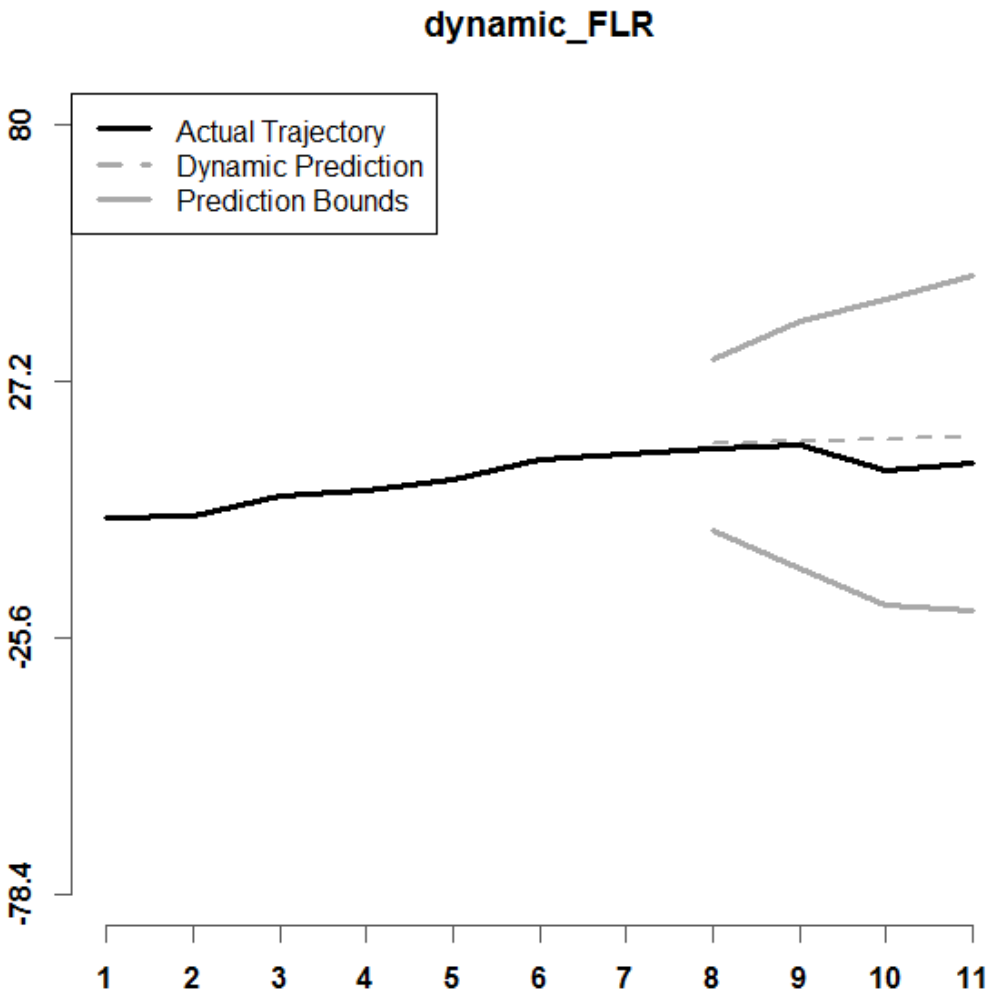

Figure 27. Shown above is a graph of a dynamic_FLR prediction interval for the Intramonthly Stock Returns for IBM for a given year, when r = 7 and C = 95% confidence.

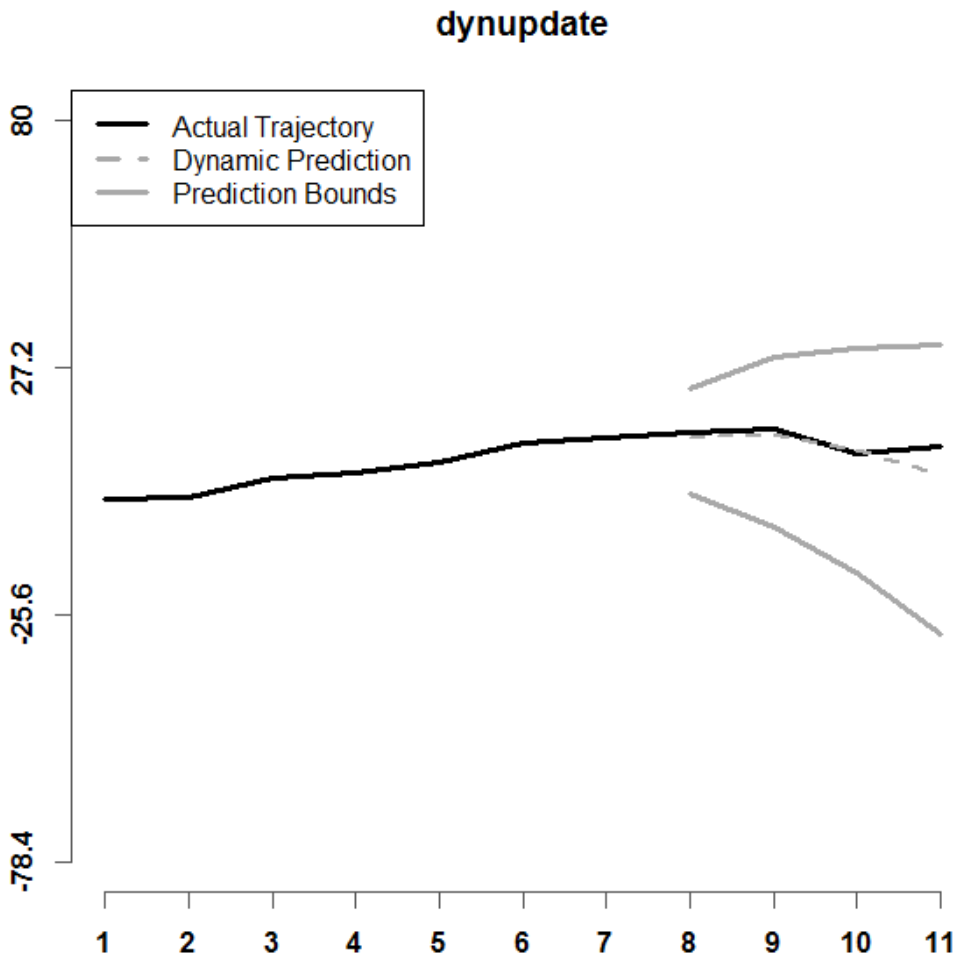

Figure 28. Shown above is a graph of a dynupdate prediction interval for the Intramonthly Stock Returns for IBM for a given year, when r = 7 and C = 95% confidence.

Figures 26 – 28 show prediction intervals for the Intramonthly Stock Returns for MSFT in the year 2000. The black lines in each panel indicate the actual stock returns. The dashed gray lines indicate the dynamic predictions that each of the methods obtained, and the solid gray lines are the lower and upper bounds of the prediction intervals by the respective methods. Note that the prediction interval for DPFFR maintains a consistent width and the actual monthly stock highs are included in the DPFFR prediction interval.

7. Conclusions

In the numerical simulations that were considered in this thesis, it appears that DPFFR is a method of dynamic prediction that outperforms several other existing methods of dynamic prediction in terms of prediction interval width, prediction interval coverage, and IMPE. Of the methods that treat the predicted response as a function, DPFFR has the lowest CPU time in most cases. In the cases where DPFFR showed similar results to other methods, DPFFR performed just as well or was very competitive in terms of the metrics. It should be noted that these conclusions only apply to the simulated cases that were considered, and further study would need to be conducted to see if these conclusions are generalizable.

When used in a practical application with data from Microsoft (MSFT) and IBM, it was observed that DPFFR had smaller IMPE than both dynamic_FLR and dynupdate, making it a preferable alternative to these methods. This is notable because all three of these methods (DPFFR, dynamic_FLR, and dynupdate) share a functional response in the model, and are similar in the regard that they only require one main iteration to predict the trajectory of the entire curve of interest. In this application, DPFFR maintained comparable coverage and prediction interval width to these existing methods, while also maintaining a coverage that was close to the nominal 95% level.

8. Further Discussion

This thesis can be viewed as a preliminary study into the performance of DPFFR as a method of creating dynamic prediction intervals. There are more simulated cases that would need to be checked before a more general conclusion can be made about the

performance of DPFFR relative to other methods of dynamic prediction. For instance, in the simulations used in this thesis, the error terms were taken to be $N(0, 0.22^2)$. Future studies could investigate what happens when the error terms have different variances, such as 0.1^2 or 0.5^2, and it would be of interest to see if the performance of DPFFR changes in these scenarios. Additional cutoff points r, other than r = 8 and r = 11, should also be considered in future studies to have a more complete assessment of the performance of DPFFR. Other forms of trigonometric and polynomial data structures for settings A and B could be studied as well. Future studies could also consider how well DPFFR performs in other real-world applications, where there are more time points, to get a better idea of the circumstances where DPFFR is reliable.

Appendix

Table A1. *Results for IMPE for 100 simulated datasets, with cutoff points r = {8,11}, number of curves n = {25, 50}, and two settings (A, B). Model (4) was used.*

	Method					
	BENDY	DLM	DPFR	DPFFR	dynamic_FLR	dynupdate
Setting A						
n = 25						
r = 8	0.15	0.16	0.19	**0.09**	0.17	0.10
r = 11	0.52	0.55	0.19	**0.07**	0.48	0.27
n = 50						
r = 8	0.14	0.07	0.13	**0.09**	0.14	0.12
r = 11	0.48	0.09	0.16	**0.06**	0.39	0.31
Setting B						
n = 25						
r = 8	0.59	0.16	0.37	**0.09**	2.34	1.61
r = 11	2.42	0.55	0.60	**0.07**	7.88	5.48
n = 50						
r = 8	0.55	0.07	0.25	**0.09**	2.01	1.71
r = 11	2.38	0.09	0.49	**0.06**	6.58	5.82

Table A2. *Simulation results for 100 simulated datasets, with cutoff points r = {8,11}, number of curves n = {25, 50}, two settings (A, B), with C = 95% confidence. Model (4) was used.*

Method	AC	AW	CPU	AC	AW	CPU
Setting A		r=8			r=11	
n = 25						
BENDY	0.95	1.58	0.97	0.94	2.84	0.57
DLM	0.95	1.76	1.38	0.95	3.97	1.00
DPFR	0.89	1.36	5.72	0.92	1.47	2.82
DPFFR	0.94	1.16	7.38	0.99	1.27	5.47
Dynamic_FLR	0.97	2.28	12.24	0.96	3.35	10.25
dynupdate	0.69	0.63	21.04	0.63	0.93	15.13
n = 50						
BENDY	0.95	1.46	1.77	0.95	2.68	1.11
DLM	0.95	1.10	2.53	0.95	1.20	1.95
DPFR	0.90	1.19	10.92	0.91	1.31	5.49
DPFFR	0.94	1.09	14.08	0.99	1.21	10.05
Dynamic_FLR	0.96	1.87	29.79	0.96	2.93	26.16
dynupdate	0.70	0.70	70.10	0.65	1.01	69.91
Setting B						
n = 25						
BENDY	0.95	3.16	0.88	0.95	6.33	0.56
DLM	0.95	1.76	1.25	0.95	3.97	0.96
DPFR	0.82	1.57	5.05	0.82	1.92	2.73
DPFFR	0.94	1.15	5.71	0.99	1.27	4.86
Dynamic_FLR	0.95	7.15	8.97	0.94	12.90	7.33
dynupdate	0.84	3.70	12.52	0.91	8.31	12.49
n = 50						
BENDY	0.95	2.95	1.78	0.95	6.13	1.12
DLM	0.95	1.10	2.49	0.95	1.20	1.93
DPFR	0.82	1.31	11.14	0.79	1.58	5.87
DPFFR	0.94	1.08	14.70	0.99	1.21	11.70
Dynamic_FLR	0.95	6.53	26.62	0.95	11.76	22.01
dynupdate	0.86	4.12	70.85	0.92	8.87	70.65

Table A3. *Simulation results for 100 simulated datasets, with cutoff points r = {8,11}, number of curves n = {25, 50}, two settings (A, B), with C = 90% confidence. Model (4) was used.*

Method	AC	AW	CPU	AC	AW	CPU
Setting A		r=8			r=11	
n = 25						
BENDY	0.90	1.31	0.94	0.90	2.35	0.60
DLM	0.90	1.42	1.35	0.90	2.94	1.02
DPFR	0.81	1.10	5.45	0.84	1.15	2.80
DPFFR	0.87	0.93	5.59	0.95	1.00	4.41
Dynamic_FLR	0.94	1.87	10.42	0.92	2.81	8.98
dynupdate	0.62	0.54	12.90	0.56	0.79	12.86
n = 50						
BENDY	0.90	1.22	1.94	0.90	2.24	1.19
DLM	0.90	0.92	2.79	0.90	1.00	2.11
DPFR	0.82	0.96	12.00	0.83	1.03	5.85
DPFFR	0.86	0.88	15.28	0.95	0.95	10.72
Dynamic_FLR	0.93	1.49	31.82	0.92	2.37	27.10
dynupdate	0.62	0.60	75.07	0.58	0.86	73.14
Setting B						
n = 25						
BENDY	0.90	2.62	1.12	0.90	5.24	0.60
DLM	0.90	1.42	1.56	0.90	2.94	1.00
DPFR	0.72	1.27	6.52	0.72	1.51	2.93
DPFFR	0.87	0.92	7.56	0.95	0.99	5.17
Dynamic_FLR	0.91	6.09	11.53	0.91	10.97	7.73
dynupdate	0.78	3.26	65.82	0.87	7.50	12.78
n = 50						
BENDY	0.90	2.46	2.00	0.90	5.11	1.19
DLM	0.90	0.92	2.85	0.90	1.00	2.06
DPFR	0.62	1.05	12.52	0.68	1.24	6.15
DPFFR	0.86	0.88	14.70	0.95	0.95	12.47
Dynamic_FLR	0.91	5.37	30.27	0.91	9.66	23.40
dynupdate	0.80	3.55	77.31	0.87	7.62	72.83

Table A4. *Simulation results for 100 simulated datasets, with cutoff points r = {8,11}, number of curves n = {25, 50}, two settings (A, B), with C = 99% confidence. Model (4) was used.*

Method	AC	AW	CPU	AC	AW	CPU
Setting A		r=8			r=11	
n = 25						
BENDY	0.99	2.16	0.75	0.99	3.87	0.48
DLM	0.99	2.56	1.09	0.99	7.29	0.83
DPFR	0.98	1.98	4.55	0.99	2.31	2.37
DPFFR	1.00	1.68	6.14	1.00	2.00	4.54
Dynamic_FLR	0.98	2.66	10.03	0.98	3.82	8.31
dynupdate	0.77	0.75	12.81	0.72	1.13	12.75
n = 50						
BENDY	0.99	1.95	1.62	0.99	3.58	1.19
DLM	0.99	1.48	2.35	0.99	1.63	2.07
DPFR	0.98	1.73	10.51	0.99	2.06	6.17
DPFFR	0.99	1.59	34.16	1.00	1.90	15.47
Dynamic_FLR	0.99	2.40	36.80	0.98	3.63	33.76
dynupdate	0.78	0.85	91.62	0.75	1.24	462.38
Setting B						
n = 25						
BENDY	0.99	4.32	0.75	0.99	8.62	0.50
DLM	0.99	2.56	1.09	0.99	7.29	0.89
DPFR	0.94	2.29	4.47	0.95	3.01	2.62
DPFFR	1.00	1.67	6.54	1.00	2.00	5.98
Dynamic_FLR	0.97	7.97	8.45	0.96	14.35	7.86
dynupdate	0.88	4.18	12.69	0.93	8.89	23.00
n = 50						
BENDY	0.99	3.94	1.68	0.99	8.18	0.96
DLM	0.99	1.48	2.49	0.99	1.63	1.69
DPFR	0.94	1.90	12.15	0.93	2.48	5.33
DPFFR	0.99	1.58	92.09	1.00	1.89	13.97
Dynamic_FLR	0.98	7.92	32.79	0.98	14.19	24.58
dynupdate	0.92	4.84	111.42	0.96	10.16	75.19

Table A5. *Results for IMPE for 100 simulated datasets, with cutoff points r = {8,11}, number of curves n = {25, 50}, and two settings (A, B). Model (1) was used.*

	Method					
	BENDY	DLM	DPFR	DPFFR	dynamic_FLR	dynupdate
Setting A						
n = 25						
r = 8	0.17	0.22	0.48	**0.09**	0.43	0.25
r = 11	0.57	3.93	0.52	**0.07**	0.74	0.43
n = 50						
r = 8	0.14	0.07	0.40	**0.09**	0.36	0.29
r = 11	0.51	0.10	0.42	**0.06**	0.62	0.49
Setting B						
n = 25						
r = 8	0.68	0.22	0.49	**0.09**	2.64	1.74
r = 11	2.66	3.93	0.59	**0.07**	8.15	6.07
n = 50						
r = 8	0.58	0.08	0.39	**0.09**	2.27	1.92
r = 11	2.50	0.10	0.47	**0.06**	6.83	6.05

Table A6. *Simulation results for 100 simulated datasets, with cutoff points r = {8,11}, number of curves n = {25, 50}, two settings (A, B), with C = 95% confidence. Model (1) was used.*

Method	AC	AW	CPU	AC	AW	CPU
Setting A		r=8			r=11	
n = 25						
BENDY	0.95	1.69	1.12	0.95	3.00	0.71
DLM	0.95	2.14	1.49	0.95	26.66	1.10
DPFR	0.76	1.65	5.21	0.81	1.89	2.98
DPFFR	0.94	1.17	5.87	0.99	1.30	4.54
Dynamic_FLR	0.96	3.32	10.46	0.95	4.02	9.03
dynupdate	0.59	0.85	12.88	0.64	1.22	12.88
n = 50						
BENDY	0.95	1.50	2.24	0.95	2.75	1.47
DLM	0.95	1.14	3.06	0.95	1.26	2.36
DPFR	0.72	1.37	12.02	0.76	1.56	6.67
DPFFR	0.94	1.10	15.68	0.99	1.22	11.74
Dynamic_FLR	0.96	2.82	30.96	0.96	3.66	29.01
dynupdate	0.59	0.91	73.24	0.65	1.31	360.27
Setting B						
n = 25						
BENDY	0.95	3.38	1.11	0.95	6.68	0.71
DLM	0.95	2.14	1.51	0.95	26.66	1.10
DPFR	0.77	1.68	4.81	0.81	1.98	2.86
DPFFR	0.94	1.16	6.11	0.99	1.29	5.22
Dynamic_FLR	0.95	7.66	8.82	0.95	13.30	7.51
dynupdate	0.86	41.12	12.94	0.91	8.54	12.93
n = 50						
BENDY	0.95	3.03	2.34	0.95	6.29	1.42
DLM	0.95	1.14	3.19	0.95	1.26	2.28
DPFR	0.73	1.38	10.92	0.77	1.61	6.34
DPFFR	0.94	1.09	17.29	0.99	1.21	13.50
Dynamic_FLR	0.95	6.86	28.65	0.95	11.92	23.57
dynupdate	0.88	4.60	383.31	0.92	9.00	74.04

Table A7. *Simulation results for 100 simulated datasets, with cutoff points r = {8,11}, number of curves n = {25, 50}, two settings (A, B), with C = 90% confidence. Model (1) was used.*

Method	AC	AW	CPU	AC	AW	CPU
Setting A		r=8			r=11	
n = 25						
BENDY	0.90	1.39	1.14	0.90	2.48	0.71
DLM	0.90	1.70	1.54	0.90	13.25	1.13
DPFR	0.66	1.33	5.30	0.70	1.48	3.00
DPFFR	0.87	0.95	5.92	0.95	1.02	4.55
Dynamic_FLR	0.93	2.76	10.53	0.91	3.43	9.10
dynupdate	0.52	0.73	12.99	0.57	1.05	12.92
n = 50						
BENDY	0.90	1.25	2.28	0.90	2.29	1.43
DLM	0.90	0.95	3.17	0.90	1.04	2.30
DPFR	0.61	1.10	12.25	0.65	1.22	6.51
DPFFR	0.86	0.88	15.89	0.95	0.95	11.16
Dynamic_FLR	0.93	2.31	31.61	0.92	2.99	27.39
dynupdate	0.52	0.78	74.32	0.58	1.11	73.30
Setting B						
n = 25						
BENDY	0.90	2.79	1.12	0.90	5.52	0.72
DLM	0.90	1.70	1.14	0.90	13.25	1.13
DPFR	0.67	1.34	4.93	0.71	1.55	2.89
DPFFR	0.87	0.94	6.21	0.95	1.01	5.27
Dynamic_FLR	0.91	6.53	8.98	0.90	11.26	7.61
dynupdate	0.80	3.64	13.04	0.87	7.69	12.94
n = 50						
BENDY	0.90	2.52	2.26	0.90	5.25	1.42
DLM	0.90	0.95	3.15	0.90	1.04	2.30
DPFR	0.63	1.11	10.54	0.65	1.26	6.29
DPFFR	0.86	0.88	16.58	0.95	0.95	13.33
Dynamic_FLR	0.91	5.64	27.64	0.92	9.85	23.40
dynupdate	0.83	3.97	73.38	0.87	7.77	73.03

Table A8. *Simulation results for 100 simulated datasets, with cutoff points r = {8,11}, number of curves n = {25, 50}, two settings (A, B), with C = 99% confidence. Model (1) was used.*

Method	AC	AW	CPU	AC	AW	CPU
Setting A		r=8			r=11	
n = 25						
BENDY	0.99	2.31	1.12	0.99	4.11	0.71
DLM	0.99	3.24	1.52	0.99	133.55	1.12
DPFR	0.91	2.40	5.21	0.96	2.96	2.97
DPFFR	1.00	1.71	5.80	1.00	2.03	4.46
Dynamic_FLR	0.98	3.78	10.37	0.97	4.53	8.94
dynupdate	0.68	1.04	12.89	0.73	1.48	12.90
n = 50						
BENDY	0.99	2.00	2.28	0.99	3.67	1.41
DLM	0.99	1.53	3.14	0.99	1.70	2.29
DPFR	0.89	1.99	12.16	0.94	2.44	6.41
DPFFR	0.99	1.60	15.87	1.00	1.91	10.96
Dynamic_FLR	0.98	3.45	31.48	0.98	4.51	26.92
dynupdate	0.69	1.12	74.48	0.74	1.60	72.95
Setting B						
n = 25						
BENDY	0.99	4.63	1.11	0.99	9.13	0.71
DLM	0.99	3.24	1.54	0.99	133.55	1.13
DPFR	0.92	2.43	4.82	0.96	3.10	2.87
DPFFR	1.00	1.69	6.08	1.00	2.02	5.20
Dynamic_FLR	0.97	8.49	8.80	0.97	14.74	7.50
dynupdate	0.90	4.62	12.91	0.93	9.14	12.91
n = 50						
BENDY	0.99	4.05	2.26	0.99	8.41	1.40
DLM	0.99	1.53	3.15	0.99	1.70	2.28
DPFR	0.90	2.00	10.50	0.94	2.52	6.23
DPFFR	0.99	1.59	16.52	1.00	1.90	13.24
Dynamic_FLR	0.98	8.40	27.61	0.98	14.39	23.07
dynupdate	0.93	5.35	73.90	0.96	10.34	72.75

References

Aneiros-Pérez, G., and Vieu, P. (2008). Nonparametric time series prediction: A semi-functional partial linear modeling. *Journal of Multivariate Analysis*, *99*(5), 834-857.

Antoniadis, A., Brossat, X., Cugliari, J., and Poggi, J. M. (2016). A prediction interval for a function-valued forecast model: Application to load forecasting. *International Journal of Forecasting*, *32*(3), 939-947.

Goldsmith, J., Bobb, J., Crainiceanu, C. M., Caffo, B., and Reich, D. (2011). Penalized functional regression. *Journal of Computational and Graphical Statistics*, *20*(4), 830-851. DOI: 10.1198/jcgs.2010.10007.

Goldsmith, J., Crainiceanu, C. M., Caffo, B., & Reich, D. (2012). Longitudinal penalized functional regression for cognitive outcomes on neuronal tract measurements. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *61*(3), 453-469.

Goldsmith, J., Greven, S., & Crainiceanu, C. I. P. R. I. A. N. (2013). Corrected confidence bands for functional data using principal components. *Biometrics*, *69*(1), 41-51.

Goldsmith, J., and Scheipl, F. (2014). Estimator selection and combination in scalar-on-function regression. *Computational Statistics & Data Analysis*, *70*, 362-372

Hyndman, R. J. and Shang, H. L. (2017), ftsa: functional time series analysis, R package, Webpage: https://cran.r-project.org/web/packages/ftsa/index.html

IBM Historical Prices, International Business Machines Stock - Yahoo Finance. (2016). Webpage: http://finance.yahoo.com/quote/IBM/history?p=IBM

Ivanescu, A. E., Staicu, A.-M., Scheipl, F. and Greven, S. (2015), Penalized function-on-function regression, *Computational Statistics*, 30(2), 539-568.

Kutner, M. H., Nachtsheim, C., and Neter, J. (2004). *Applied Linear Regression Models.* Boston, McGraw-Hill/Irwin.

MSFT Historical Prices, Microsoft Corporation Stock - Yahoo Finance. (2016). Webpage: http://finance.yahoo.com/quote/MSFT/history?p=MSFT

R Core Team (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. Webpage: *https://www.R-project.org/*

Ramsay, J. O., and Silverman, B. W. (2005). *Functional Data Analysis*. New York, Springer.

Ramsay, J. O., Wickham, H., Graves, S., & Hooker, G. (2017). fda: Functional Data Analysis. R package version 2.4.4. Webpage: *https://CRAN.R-project.org/package=fda*

Shang, H. L. (2015). Forecasting Intraday S&P 500 Index Returns: A Functional Time Series Approach. Webpage: *https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2647233*

Shang, H. L., and Hyndman, R. J. (2016). *Grouped functional time series forecasting: An application to age-specific mortality rates* (No. 4/16). Monash University,

Department of Econometrics and Business Statistics. Webpage: *http://business.monash.edu/econometrics-and-business-statistics/research/publications/ebs/wp04-16.pdf*

Wood, S.N. (2006). *Generalized Additive Models: An Introduction with R.* Chapman and Hall.