Chapter 13
Financial Trading

Investment strategies have long been the cornerstone of the financial industry. Indeed, the economic health of industrial sectors and nations is often gauged by their performance in various global stock market trading centers (e.g., the New York Stock Exchange). The volatility of stock portfolios is especially interesting in financial trading as high volatility, or strong variance in stock prices over time, is the basis for formulating buy-side and/or sell-side strategies. Specifically, formulating a principled strategy for shorting a stock (selling) or going long with a stock (buying) is critical for successfully capitalizing on market trends and generating strong return on investments.

The past few decades have seen a revolution in how and when financial trading strategies are implemented. This revolution was generated by the computer-enabled rise of algorithmic trading schemes. As of 2006, one third of all European Union and United States stock trades were estimated to be executed from a computer trading algorithm [139, 281]. The London Stock Exchange trading during this time was estimated to be 40% driven by algorithmic trading. Since 2006, and thus only over the past decade, the financial industry has seen the advent and dominance of high-frequency trading schemes. High-frequency trading attempts to take advantage of stock price fluctuations that occur on time scales that are as fast as microseconds. In 2009, high-frequency trades were estimated to accounted for 60–73% of all United States equity trading volume [139, 281]. These high-frequency trading methods can only be executed by computer, leading to a further confluence of automated traders and innovations in mathematical algorithms.

13.1 ▪ Financial investment and algorithmic trading

Algorithmic trading is driven by mathematical models, statistical estimation, and modern data-driven (predictive) analytics, which seek to capitalize on stock market patterns, either real or perceived, to inform automated buy/sell/hold investment decisions. As one might expect, the underlying mathematical tools used for algorithmic trading capitalize on a variety of statistical and probabilistic computational tools for processing the time series of financial data streams. Most of these mathematical tools are also used to generate risk assessments and confidence intervals for a given strategy; i.e., in addition to a trading decision, a prediction of volatility is often generated by the underlying statistical model used.

Thus, at its core, the ultimate goal of algorithmic trading is to take advantage of the speed and precision of computer-executed trading strategies that are based on a principled, underlying mathematical or statistical structure in financial data. In the high-frequency setting, the computer can evaluate and execute trades on time scales that are impossible for human traders to achieve. A vast number of mathematical and statistical strategies have been suggested and implemented in different markets and for a variety of financial instruments. Many of these are reviewed in standard textbooks on finance and trading [109, 3], with risk assessments of such strategies a critical factor in their ultimate viability [72, 276]. A few of the most common trading models take advantage of time-series analysis [38, 65, 268], trend following [83, 126, 259], technical indicators [66, 258], pairs trading [90], mean reversion [179, 75, 10], Markov models [144, 99], Bayesian inference [8, 241], news and announcements [305, 312], and potentially combinations of these methods [88, 34, 256]. For instance, in trend following, the trading strategy follows the trends in moving averages, channel breakouts, price-level movements, and other related financial indicators [83, 126, 259]. These are perhaps the simplest strategies to implement through algorithmic trading, since they do not involve price forecasting. Trades are simply executed based on the occurrence of desirable trends. In contrast, in an algorithm like *pairs trading*, the strategy monitors performance of two historically correlated securities [90]. When the correlation between the two securities temporarily weakens, i.e., one stock moves up while the other moves down, the pairs trade would be to short the outperforming stock and to long the underperforming one, betting that the spread between the two would eventually converge. The divergence within a pair can be caused by temporary supply/demand changes, large buy/sell orders for one security, reaction to important news about one of the companies, and many other causes. Of course, this is only a small sampling of methods and references, and we encourage the reader to consult texts on financial trading for a broader viewpoint and additional references [109, 3, 72, 276].

Common to all the trading strategies is the use of financial time-series data to make principled decisions about investment. The strategies range from simple assessments of the data to sophisticated statistical analysis of data streams. In each case, a philosophical viewpoint is taken about the meaning and interpretation of the data. In accordance with this observation, the viewpoint advocated by Mann and Kutz [189] assumes the stock market to be a complex, dynamical system that exhibits nonstationary, multiscale phenomena. As is advocated here, such a viewpoint is highly appropriate for a method like DMD and/or many of its variants, such as mrDMD. Most important, the DMD method does not enforce or prescribe an underlying dynamical model. Rather, the *equation-free* nature of the method allows the dynamics to be reconstructed directly from the data sampled over a specified window of time. Specifically, DMD decomposes stock portfolio data into low-rank features that behave with a prescribed temporal dynamics. In the process, the least-square fit linear dynamical system allows one to predict short-time future states of the system. These short-time predictions can be used to formulate successful trading strategies by identifying transient, evanescent signals in the financial data [189]. The method is adaptive, updating its decomposition as the market changes in time. It can also implement a learning (machine learning) algorithm to track optimal sampling and prediction windows, as these change much more slowly in time and in different market sectors. The method is applied and demonstrated on several markets (e.g., technology, biotech, transport, banks). As outlined in the introduction, DMD can also be used as a diagnostic tool, allowing for a principled technique capable of data-driven discovery of cyclic spatiotemporal features in a given data set. Indeed, the DMD method has recently been used by Hua et al. [137] to extract and

analyze robust economic cycles in stock market time-series data. Their algorithm considers an innovative approach to separating time-series data into robust and nonrobust features to observe persistent cyclic activity.

We demonstrate the use of DMD for a financial trading strategy [189]. We specifically consider taking advantage of the growth and decay behavior of the decomposition as the critical features, not necessarily the cyclic behavior reported by Hua et al. [137]. The DMD algorithm also allows one to adapt the frequency and duration (sampling window) of the market data collection to sift out information at different time scales, making different trading strategies (e.g., high frequency trading, daily trading, long-term trading, etc.) possible. Indeed, one can use an iterative refinement process to optimize the snapshot sampling window for predicting the future market. A critical innovation of DMD is its ability to handle transient phenomena and nonstationary data, which are typically weaknesses of SVD-based techniques. One can also build upon recent innovations in mrDMD to mine for data features at different time scales [167].

13.2 ▪ Financial time-series data and DMD

Aside from small, open-source exemplar data sets, financial data is rarely free. Well-curated and scrubbed data is particularly valuable and difficult to obtain without paying a fee. With this in mind, we demonstrate the DMD algorithm on a readily available data source provided by Yahoo! web services. Specifically, we consider daily stock price quotes. For intraday trading or high-frequency data streams, other sources of data should be considered. Regardless, the efficacy of the DMD algorithm can be demonstrated on daily stock quotes.

MATLAB allows for pulling daily stock quotes directly from Yahoo! using the *data feed* toolbox. Thus, the data can be imported to the MATLAB framework for processing and evaluation. The following code demonstrates how this is done in practice.

***ALGORITHM* 13.1. Yahoo data feed.**

```
c = yahoo;
d = fetch(c,'IBM');
```

This pulls stock data information for the technological giant IBM. Specifically, the retrieved data in the structure **d** is as follows.

***ALGORITHM* 13.2. Yahoo data options.**

```
d =

    Symbol: {'IBM'}
      Last: 173.84
      Date: 735529.00
      Time: 0.42
    Change: 0.98
      Open: 173.23
      High: 173.84
       Low: 172.95
    Volume: 1132526.00
```

This then allows one to acquire important information pertaining to, for instance, the current date and time along with the daily high and low. One can also pull the current value of the stock with the fetch command.

***ALGORITHM* 13.3. Current stock price.**

```
d = fetch(c,'IBM',now)

d =

735528.0 174.42 174.75 172.63 172.86 7079500.0 172.86
```

This version of fetch returns the date, open price, high price, low price, closing price, volume, and adjusted closing price. For our purposes, we would like to extract the closing price for a series of stocks over a prescribed time period. To pull data over a range of times, the following can be used.

***ALGORITHM* 13.4. Stock price history.**

```
ClosePrice = fetch(c,'IBM','Close','08/01/99','08/25/99')

ClosePrice =

     730357.00         122.37
     730356.00         122.00
     730355.00         124.44
     730352.00         121.75
     730351.00         122.94
    ...
```

This gives the closing price over a period specified by the user. One could also pull opening prices or highs and lows. In what follows, we will specify a number of stocks traded over a prescribed period of time. The following gives the form of command used for pulling our data from Yahoo!.

***ALGORITHM* 13.5. Yahoo daily stock data.**

```
tickers= {'JPM' 'BOA' 'AMZN' 'CVX'};

date_1 = 'Jun 1 06';
date_2 = 'May 10 13';

for j = 1:length(tickers);
    Price.(tickers{j})= fetch(yahoo,tickers(j),'Adj Close',
        date_1, date_2, 'd');
    Temp = Price.(tickers{j});
    ClosePrice(:,j) = Temp(:,2);
end
Close_Price = flipud(ClosePrice);
```

Importantly, one only need specify the stock ticker symbol as used by the New York Stock Exchange or NASDAQ, along with the start and end dates required by the data

pull. The stock ticker symbol is the simple identifier for all publicly traded companies. For instance, in the banking sector, 'JPM' and 'BOA' are J. P. Morgan Bank and Bank of America, respectively. In the retail sector, 'AMZN' and 'CVX' are Amazon and CVX pharmacies, respectively. Start and end prices are given for month (abbreviated), day, and year (last two digits). In the algorithm presented, the daily closing price of the stock is extracted. The opening price can also be acquired if desired.

Once the daily stock prices have been collected, the data can be arranged into a data matrix for processing. Specifically, the following data matrix is constructed:

$$\mathbf{X} = \begin{bmatrix} \cdots & \vdots & \cdots \\ \cdots & \text{'JPM'} & \cdots \\ \cdots & \text{'BOA'} & \cdots \\ \cdots & \text{'AMZN'} & \cdots \\ \cdots & \text{'CVX'} & \cdots \\ \cdots & \vdots & \cdots \end{bmatrix} \quad \downarrow \text{companies}$$

$$\longrightarrow$$

daily stock prices

where the daily stock price sequences are arranged chronologically from the first day to the last day pulled in the data extraction. Note that the time differential between the columns is important in the DMD process. With a prescribed differential of one day (for daily stock data), the algorithm is easy to execute given the equally spaced sampling.

The DMD method provides a decomposition of the collected data into a set of dynamic modes that are derived from stock market snapshots over a given time period. As shown in the data matrix $\mathbf{X}$, the data collection process involves two parameters:

$$n = \text{number of companies in a given portfolio,}$$
$$m = \text{number of data snapshots taken.}$$

With this data matrix, we can evaluate portfolios of holdings and different financial sectors, and we can determine how to best use portions of the overall data matrix $\mathbf{X}$. Note that the data matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$ is as defined for the DMD algorithm in (1.16).

The goal of a DMD-based trading algorithm is to capitalize on the predictions (1.24) of the DMD theory. The trading algorithm developed is parameterized by two key (integer) parameters:

$$m_p = \text{number of past days of market snapshot data taken,}$$
$$m_f = \text{number of days in the future predicted.}$$

Specifically, we will refer to the DMD prediction (1.24) with the notation

$$\mathbf{x}_{\text{DMD}}(m_p, m_f) \tag{13.1}$$

to indicate the past m_p number of days that are used to predict m_f days in the future. This allows us to specify both the market sampling window and how far into the future we are predicting. Note that the DMD prediction $\mathbf{x}_{\text{DMD}}(m_p, m_f)$ uses only portions of the full data matrix $\mathbf{X}$ as specified by the parameter set (m_p, m_f). Our objective is to use historical data to determine suitable combinations of (m_p, m_f) that give the best

predictive value. In particular, we look for what we term *trading hot spots*, or regions of (m_p, m_f) where the predictions are optimal. A more precise definition of a trading hot spot will be given in what follows.

13.3 ▪ Trading algorithms and training

The training algorithm is essentially an optimization routine over the space of parameters (m_p, m_f). In particular, our objective is to see which DMD predictions $\mathbf{x}_{\text{DMD}}(m_p, m_f)$ have the highest rate of success in predicting the future position of the market, or portfolio of holdings. The training portion of the algorithm thus looks over a historic time period, whether that is 100 days or 10 years, to determine the best choices of (m_p, m_f). This is an *off-line* procedure run on a collection of historical data. By learning the best (m_p, m_f), it may be possible to execute trading in an *on-line* fashion since this potentially expensive optimization procedure has already been performed.

We consider all possible combinations of (m_p, m_f) and their associated success rates for predicting the correct future state, whether the stock values have increased or decreased. Since we are using historical data, we can compare the DMD prediction with known market activity. Specifically, we evaluate whether DMD predicts that the market increases or decreases, and we compare that to the known market activity. We set limits that were found to be suitable for the DMD algorithm, letting $m_p = 1, 2, \ldots, 25$ days and allowing $m_f = 1, 2, \ldots, 10$ days. As will be shown in the results, these appear to be reasonable and effective values for determining the best combinations of (m_p, m_f).

When we looked at the training algorithm over the last 10 years, we could see consistent trading hot spots across sectors. Most hot spots would look at the last 8 to 10 days of prices to make the best prediction of the price in 4 to 5 days. Hot spots that had success rates greater than 50% were promising because they would be statistically likely to make money over time. The information gathered about hot spots allowed us to create a trading algorithm that would enter stock market positions using results from DMD analysis each day and the solution (13.1). In practice, the optimal values of (m_p, m_f) were determined from a two-year period of trading (July 2005 to July 2007). This training period then allowed us to find the optimal DMD prediction $\mathbf{x}_{\text{DMD}}(m_p, m_f)$ for executing trades. Thus, the training data is separate from the test set; i.e., the test set is from July 2007 onwards. Note that in addition to determining successful prediction windows, an assessment of the net positive or negative gain is also noted. Provided there is sufficient predicted change in the stock (at least 1%), then a trade is executed. It is critical to offset the cost of the transaction fee to make money.

In the results that follow, three basic assumptions are made: (i) the initial investment capital is \$1 million, (ii) transaction costs are \$8 for each position, and (iii) all money is invested evenly across all companies in a given portfolio. As such, the method used is characterized as a self-financing strategy as there is no exogenous infusion or withdrawal of money; the purchase of a new asset must be financed by the sale of an old one. We had the flexibility to use any company, providing it had been publicly trading for the time frame we were using, and we were able to combine as many companies as we wished. For illustrative purposes, we tended to use 10 companies as a proxy, or representation, of each sector considered. The initial daily trading algorithm took given inputs (m_p, m_f) for the sampling window and prediction window and ran the DMD

analysis each day. Specifically, trading was performed using the DMD hot spot prediction windows and capital was divided equally among all companies in the portfolio. After we entered the position for a given duration, we calculated how much money would have been made or lost by using historical data. We also reinvested gains over the duration of the trade period. After the algorithm had completed, we compared our results to buying the benchmark, S&P 500, and also compared it to buying and holding the individual stocks. Note that effects of slippage, for instance, have been ignored in the trading scheme. However, the $8 per trade is a conservative (high) estimate of trading costs that should offset such effects. More precisely, it should be noted that the trading fee of $8 is selected from the base fee of a company like E-trade, which charges $7.99 per trade. Thus, one would not expect to pay anything higher than this rate. However, *Barron's Annual Best Online Brokers* survey [136] evaluates a number of brokers for various characteristics, including the transaction fee per trade. This number can be highly variable depending on the size of brokerage and stocks being traded. For instance, Interactive Brokers charges a nominal fee of as low as $1 per trade. Such rates were effective as of November 4, 2015. Many of the firms listed in the ranking typically had additional fees, and some firms reduced or waived commissions or fees, depending on account activity or total account value. Thus, setting a precise cost for trading is difficult.

The second trading algorithm we created did not use any information that one wouldn't have if one wanted to trade today; hence it was as realistic as possible. The algorithm would start trading at day 101 because it would continuously use the previous 100 days to find the optimal trading inputs from the training algorithm. Therefore, the training algorithm would be used on a sliding 100-day window prior to the day the trading algorithm was executed. It would update its results daily using the previous 100 days. Throughout our research we found that most sectors had obvious, even prominent, hot spots. However, some sectors didn't have any clear hot spot, and they tended to be the sectors that underperformed in the trading algorithm.

With this in mind, we created a third trading algorithm that looked into whether the inputs with the maximum success rate were within a larger hot spot region or isolated instances and likely to have performed well over the last 100 days due to randomness. To do this, the trading algorithm found out what inputs had the maximum success rate over the last 100 days, and then it looked at the surrounding inputs to see if the mean success rate of all nine neighboring (m_p, m_f) were above a threshold. The 100-day window was chosen empirically. Our objective was to construct a test window that was sufficiently long to capture the recent trends, but not too long to be weighted by data long into the past. Thus, a very long sampling window made the method extremely slow to adapt due to the need to respect the distant past of the financial time series, whereas a short sampling window made the adaptability very volatile. Given our typical $(8,5)$ trading window, the 100-day window nicely balanced historical data with adaptability. If a hot spot region was found, then it would be classified as a hot spot and a trade would be executed. Otherwise, the trading algorithm would hold the money until a hot spot appeared at a later date. When implementing this strategy, we used a hot spot threshold of 53% so that we could be confident that there truly was a hot spot. This is perhaps the most robust of the trading strategies, as we demonstrated that markets with hot spot regions were quite amenable to the DMD strategy. Indeed, all market sectors showing a strong hot spot generated significant capital gains with the trading algorithm.

In summary, three algorithms based on DMD can be easily developed:

(i) Algorithm I: We train on historical financial data over a two-year period (June 2005 to June 2007) and then trade on the best discovered (m_p, m_f) sampling and prediction window from June 2007 onwards. Thus, there is a two-year training period, and no out-of-sample data is used to determine the optimal window. We note that under cross-validation analysis, the optimal sampling and prediction window (m_p, m_f) is robust to changes in the two-year training period.

(ii) Algorithm II: We use the previous 100-day window to find the best (m_p, m_f) sampling and prediction window. The window is adaptively updated with the previous 100 days and we always trade based on the best prediction. We note that under cross-validation analysis, the optimal sampling and prediction window (m_p, m_f) is robust to changes in the 100-day training period. However, shorter sampling periods can induce strong variability in the results and should be avoided, while longer periods lead to the results shown in Algorithm I.

(iii) Algorithm III: This is the same as Algorithm II except that we only trade if a hot spot (13.2) (see below) is found.

In what follows, only the first algorithm is demonstrated on data. For details regarding the other strategies, see Mann and Kutz [189].

13.3.1 ▪ Trading hot spots

In general, one is interested in the highest probability of prediction success possible. In practice, for financial trading, it is difficult to achieve performance success much above 50% due to the volatile fluctuations in the market. Thus far we have been using the term *hot spot* loosely. Here, we more precisely define the concept. Numerical simulations on training data show that trading success with DMD depends not just on the best probability of prediction, but rather on having a cluster of (m_p, m_f) values where strong predictive power is achieved [189].

More precisely, we define a hot spot as a particular pairing of the integers (m_p, m_f) such that (i) the prediction accuracy is above 53% at that location and (ii) the average prediction score of this location and its eight neighboring cells averages a prediction value of 53% or above. Denoting the success rate at the location (m_p, m_f) as S_{m_p,m_f}, we have the hot spot conditions

$$\text{(i) } S_{m_p,m_f} > 0.53, \quad \text{(ii) } \frac{1}{9}\sum_{j=-1}^{1}\sum_{k=-1}^{1} S_{m_p+k,m_f+j} > 0.53. \tag{13.2}$$

The value of 0.53 is chosen from empirical observations of the predictive success of the DMD algorithm. In what follows, we focus on two key steps: (i) a training step in the algorithm to determine (m_p, m_f) and (ii) implementation of trading based on the results. If there exists a trading hot spot with the above definition, then the DMD algorithm provides a highly effective method for algorithm trading.

13.3.2 ▪ Trading hot spots by sector

Before proceeding to evaluate the performance of the DMD algorithm for algorithm trading, we consider various financial sectors and their potential for producing a hot

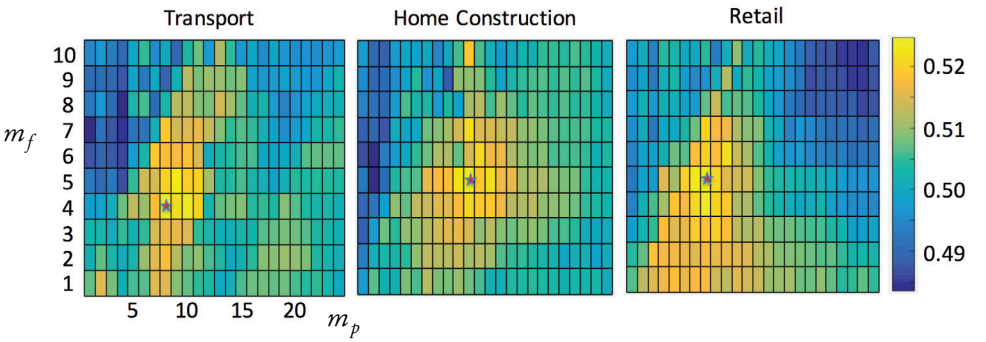

Figure 13.1. *Sectors that have a hot spot where the probability of success at the optimal trading position for* (m_p, m_f) *is above* 53% *along with its surrounding eight cells. Thus, the definition* (13.2) *is satisfied in the transport, home construction, and retail sectors. Note that the color in each cell denotes the probability of success.*

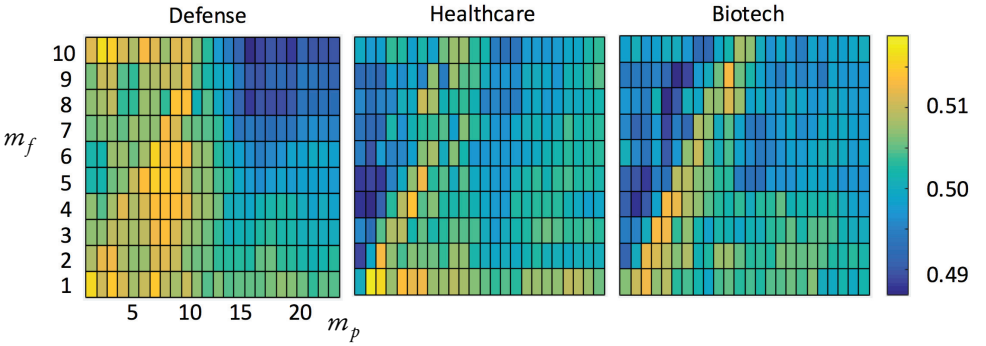

Figure 13.2. *Sectors that do not have a hot spot. Although individual cells may be above* 53% *in predictive accuracy, these sectors fail to have the surrounding eight cells above the threshold in* (13.2)*. These sectors include defense, healthcare, and biotech. Note that the color in each cell denotes the probability of success.*

spot. Six examples are considered in total. In the transport, home construction, and retail sectors, a trading hot spot as defined above is produced. In contrast, the sectors of defense, healthcare, and biotech fail to produce a hot spot as defined. When a hot spot is produced, DMD produces a viable and effective trading scheme. When no hot spot is present, DMD fails to produce results that are any better than simply investing in the S&P 500 and holding. Thus, the definition of a trading hot spot serves a practical purpose in defining when the DMD trading algorithm might work.

Figures 13.1 and 13.2 show the success rates for the DMD algorithm when compared to the financial market fluctuations. These results are computed on historical data to find the best trading windows (m_p, m_f) that maximize the success rate. In the sectors where a hot spot as defined in (13.2) is found, the trading performance will be demonstrated to be successful. If a hot spot as defined by (13.2) fails to exist, then the DMD algorithm statistically does no better than investing and holding in the S&P 500. In the transport, home construction, and retail sectors of Figure 13.1, clear hot spots are found, with (m_p, m_f) pairings of $(8,4)$, $(11,5)$, and $(8,5)$, respectively. The defense, healthcare, and biotech sectors have no hot spots. Note that for each sector we select a small, representative set of companies to illustrate that sector's performance.

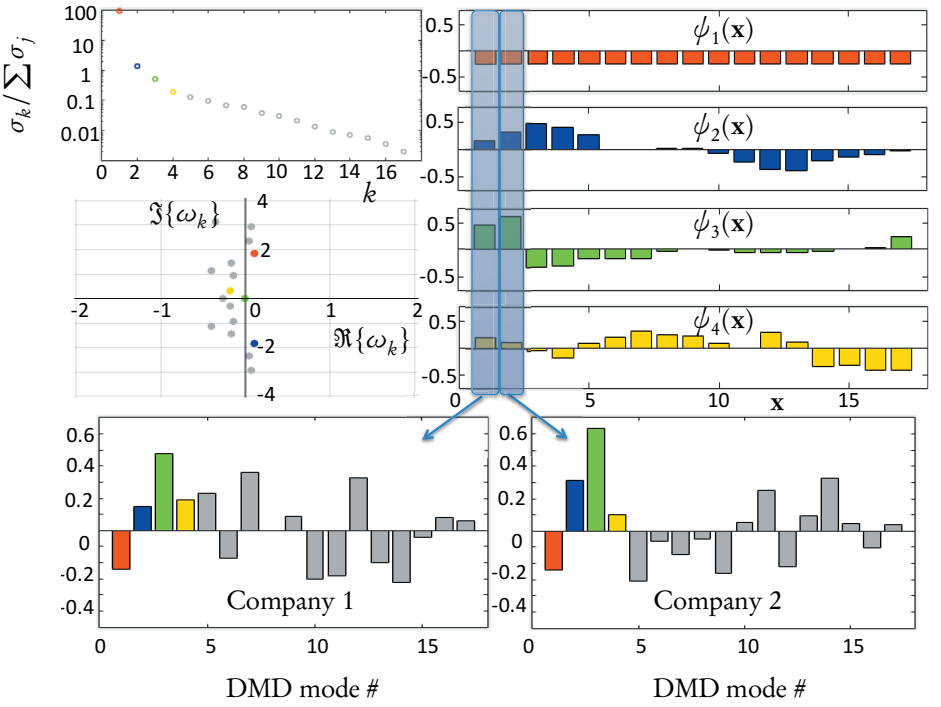

Figure 13.3. *DMD of an* 18*-day sampling of portfolio data in the biotech and healthcare sector (*17 *companies): 'DHI' 'LEN' 'PHM' 'TOL' 'NVR' 'HD' 'LOW' 'SHW' 'ONCS' 'BIIB' 'AMGN' 'CELG' 'GILD' 'REGN' 'VRTX' 'ALXN' 'ILMN'. The top left panel shows, on a log scale, the percentage of information captured in each mode from the SVD* (1.18) *($\sigma_j / \sum \sigma_k$, where σ_k are the diagonal elements of Σ). The data, which is color coded throughout the figure, is shown to be dominated by a few leading modes (colored red, blue, green, and yellow in order of variance contained). The middle left panel shows the* 17 *eigenvalues ω_k of each mode used in the solution* (1.24)*. Eigenvalues with $\Re\{\omega_i\} > 0$ represent growth modes. The four top right panels show the leading DMD modes ($\psi_k(\mathbf{x})$) and their composition from the* 17 *companies selected. The first mode (red) shows the "background" portfolio, or average price of stocks, over the sampling window. The decomposition of the first and second companies on the* 17 *DMD modes is shown in the bottom two panels, where the first four modes are highlighted. Reprinted with permission from Taylor and Francis* [189].

13.4 ▪ Trading performance

With the definition of a hot spot, the DMD-based trading strategy can be executed and evaluated. Before assessing its performance on trading, it is instructive to first illustrate DMD. Consider a sample of 17 companies over an 18-day trading window from the biotech and healthcare sectors: 'DHI' 'LEN' 'PHM' 'TOL' 'NVR' 'HD' 'LOW' 'SHW' 'ONCS' 'BIIB' 'AMGN' 'CELG' 'GILD' 'REGN' 'VRTX' 'ALXN' 'ILMN'. Figure 13.3 shows all aspects of the resulting decomposition. Specifically, the SVD produces a diagonal matrix whose entries determine the modes of maximal variance. In this case, there is a low-rank structure to the data, as demonstrated in Figure 13.3. The first mode is particularly important, as it represents the average price across the sampling window. The first four DMD modes, which are composed of weightings of the 17 companies, are highlighted, as they are the most dominant structures in the

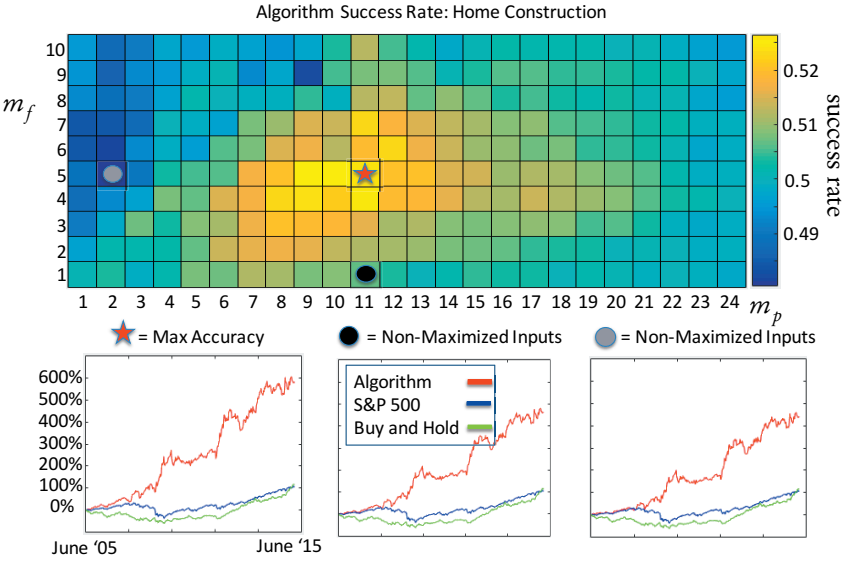

Figure 13.4. *Success rates achieved using different sampling and prediction windows* (m_p, m_f) *for the DMD algorithm when back-testing. The hot spot (top panel) shows a success rate of about* 52.5% *over the last* 10 *years, meaning that* 52.5% *of all the trades we executed were correct and made money. The bottom panels show how much money the algorithm makes if we use different inputs for DMD in comparison to the S&P* 500 *as well as a buy-and-hold on the stocks used in the DMD algorithm. Using the best hot spot* $\mathbf{x}_{DMD}(11,5)$ *gives the best return of* 21.48% *annualized over* 10 *years; however, using other inputs, such as* $\mathbf{x}_{DMD}(11,1)$ *or* $\mathbf{x}_{DMD}(2,5)$*, we still get promising results of* 19.22% *and* 18.59% *annualized, respectively. When calculating the money we made, we ran the DMD and traded off its signal each day, entering a position, either long or short, on every company in the algorithm. In the case above we used nine companies in the home construction sector ('DHI' 'LEN' 'PHM' 'TOL' 'NVR' 'HD' 'LOW' 'SHW' 'SPY'). Reprinted with permission from Taylor and Francis* [189].

data. The distribution of eigenvalues ω_k is also illustrated, showing the modes that have growth, decay, and/or oscillatory behavior. The DMD scheme takes advantage of identifying the largest growth modes for investment purposes. Finally, the weighting of each company on the DMD modes is also illustrated. This is the information extracted at each pass of the DMD algorithm for a given data sampling window.

DMD can be used for trading by taking advantage of identifiable hot spots in $\mathbf{x}_{\text{DMD}}(m_p, m_f)$. As already outlined in the learning algorithm, the optimal m_p and m_f can be identified and investments made accordingly. As a specific example, Figure 13.4 shows the hot spot generated in the home construction sector. The hot spot has a success rate of 53% or greater over the last 10 years, meaning that 53% of all the trades we executed were correct and made money. The bottom panels in this figure show how much money the algorithm makes if we use different inputs for DMD. Using the best hot spot, $\mathbf{x}_{\text{DMD}}(11,5)$, gives the best return of 21.48% annualized over 10 years; however, using other inputs, such as $\mathbf{x}_{\text{DMD}}(11,1)$ or $\mathbf{x}_{\text{DMD}}(2,5)$, we still get promising results of 19.22% and 18.59% annualized, respectively. We calculated the money earned

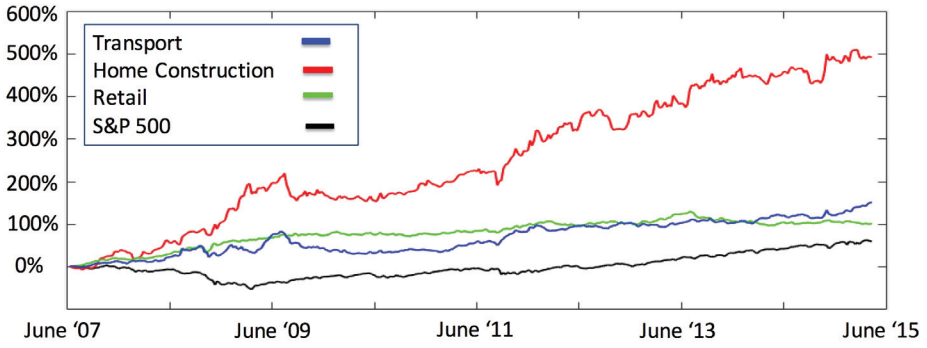

Figure 13.5. *Trading success of Algorithm I of the DMD method for three sectors exhibiting hot spots (transport, home construction, and retail). The optimal values of* (m_p, m_f) *are determined during a training period of two years (July* 2005 *to July* 2007*). Once determined, the hot spot trading window is used to execute trades from July* 2007 *onwards. When a hot spot is present, the algorithm can significantly increase financial gains over investing and holding in the S&P* 500*.*

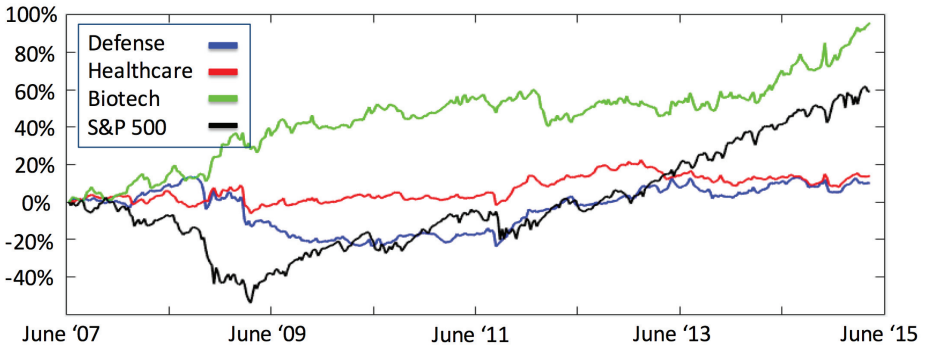

Figure 13.6. *Demonstration of trading success of Algorithm I of the DMD method for three sectors that do not exhibit a hot spot (defense, healthcare, and biotech). The optimal value of* (m_p, m_f) *is still determined during a training period of two years (July* 2005 *to July* 2007*). Once determined, the trading window is used to execute trades from July* 2007 *onwards. When there is no hot spot present, the algorithm performs no better than investing and holding in the S&P* 500*.*

from the DMD algorithm by trading from its predictions each day, entering a position, either long or short, on every company in the algorithm. In the case above we used nine companies in the home construction sector ('DHI' 'LEN' 'PHM' 'TOL' 'NVR' 'HD' 'LOW' 'SHW' 'SPY'). The home construction industry is interesting, since even when trading off the optimal, money is generated. This is not common for the DMD algorithm and is more reflective of the home construction sector growth during this period.

To more accurately assess the DMD trading strategy, consider Figures 13.5 and 13.6. The first of these figures shows the performance of DMD trading when a hot spot is present during the training period of July 2005 to July 2007 (see Figure 13.1). From July 2007, the optimal (m_p, m_f) is used for trading. With a hot spot, the DMD algorithm provides a viable trading strategy, performing well above holding stock in the S&P 500. If no hot spot is present during training (see Figure 13.2), trading is executed on the optimal (m_p, m_f), with mixed results achieved. Figure 13.6 shows the results for three sectors that fail to exhibit a hot spot. In this case, it is unclear whether the algorithm should be used instead of simply investing money in the S&P 500.