Highlights from "Cycles Discovery: Metrics, Sparsity, and Needles in a Haystack"

Michael M. Forbes, Hossein Noorazar, Gary Sandine, and Kevin R. Vixie

April 18, 2021

When looking for cycles in data, Fourier techniques are often the tools of choice. The are efficient, and allow one to quickly identify the dominant periodic trends. However, they have several limitations which we resolve.

In particular, Fourier techniques are difficult to apply when data is irregular or missing, and they are of no use when the underlying patterns are not periodic cycles. Our paper demonstrates how to use a **dictionary** to efficiently find patterns that govern underlying data. Once identified, these patterns can be used to compress that data, identify important features, and predict future behavior.

We start with the example (Section 4.3 of the paper) of the maximum daily temperature in Moab, Utah. Here we expect an annual cycle. Given complete and regularly spaced data, Fourier techniques work well, but what happens if the data is incomplete and irregularly spaced? E.g., try to find the annual cycle from the 100 randomly selected dates shown in the top of Fig. 1: The cycles.org analyzer interpolates between missing data, introducing significant errors.

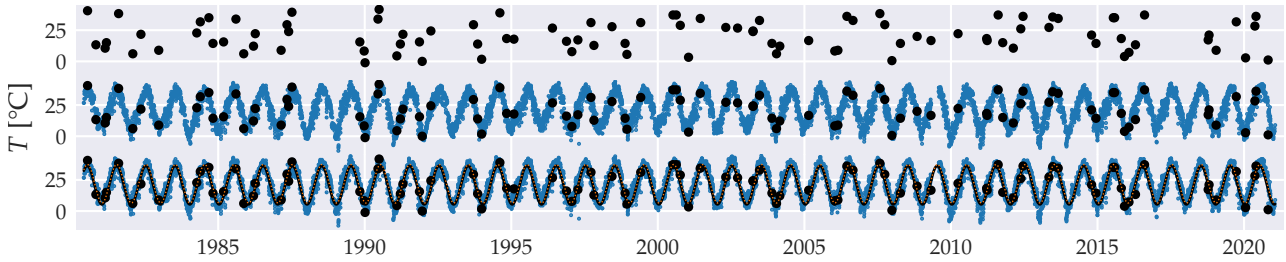


Figure 1: Maximum daily temperatures in Moab, Utah. The top panel shows $T = T_{\max}$ at 100 randomly selected days over 40 years showing almost no discernible cycles. Simply knowing that a cycle should exist allows us to extract the period of the tropical year to within about 5 hours (black cycle in the bottom panel). Adding more data (small blue dots in the middle panel) makes the annual cycle apparent and allows us to improve the precision to within 50 mins, but is not needed to find the cycle. (The dotted orange line in the bottom plot shows the cycle as extracted with 10 000 data points. This essentially lies on top of the cycle extracted from only 100 points.)

Our approach is to build a dictionary of cycles[1] for a large set of periods T. We then search the dictionary for the entries with the best match. *(This is the **orthogonal matching pursuit** algorithm described in Section 4.1).* We can also include cycles with arbitrary periods T to form an **overcomplete dictionary**: one that has many more entries than those accessible to Fourier techniques which are limited by the length of the signal, and the sample frequency. This allows us to match underlying signals with high accuracy.

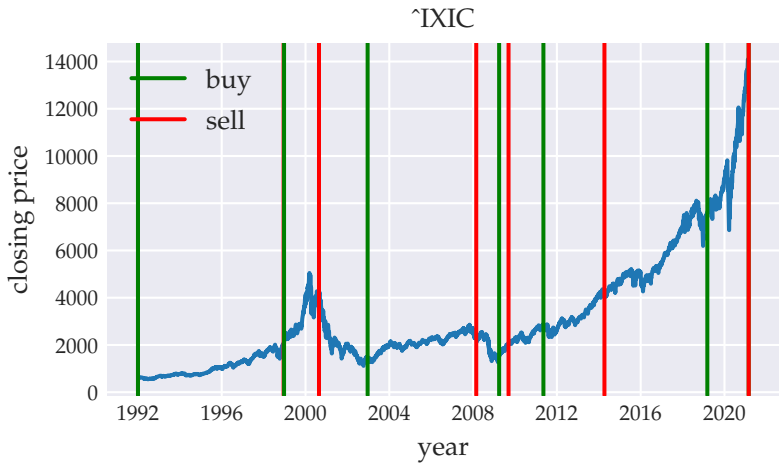
With this approach, the 100 points in the top panel of Fig. 1, which have no visually apparent pattern, gives a best matching period of $T = 365.2(2)$ days.[2] Including more data makes the cycle more visible, and improves the accuracy of the period extraction $T = 365.24(2)$ days, corresponding to the *tropical year* of 365.242 days.

[1] The dictionary entries are simply the target pattern $y = \cos(2\pi t/T + \phi)$ evaluated at the available dates t. There is no need for interpolation, and the method can be trivially generalized to skewed cycles, or non-periodic patters, by changing the functional form.

[2] The error ± 5 hour error is estimated by repeating the process with different random sets of 100 dates.

The paper discusses additional details, such as how to determine the frequency spacing in the dictionary, but the essence is simple: Use knowledge about the form of the underlying patterns[3] to find the best match. In the paper, we refer to this as finding a **sparse representation** of the data: what are the few most important patterns that best describe the data? If one knows the form of these, then one can us this technique to find a good set of patterns from the data,[4] and then use these to make predictions.

If such knowledge is not available, one may turn to the data itself to see if it contains any intrinsic patterns.[5] In this case, the appropriate patterns are almost certainly *not* going to be simple cycles, and Fourier methods will struggle. The approach of finding sparsity with overcomplete dictionaries, however, remains as easy to implement for non-cyclic patterns as it does for cyclic ones.



As another example, we give a straightforward proof of concept with a one-stock-at-a-time analysis of the Nasdaq composite (^IXIC) from 1992 through 2021. Fig. 2 shows the full history of the strategy, and examples of sell and buy signals show in Fig. 3.

Our model has many parameters, including the lengths of the training and prediction windows, the number of cycles used to model the historical data, and the number of cycles that must coincide before triggering a buy or sell. These methods using dynamic combinations of parameters to identify specific trends could be useful in combination with other methods for estimating the likelihood of upturns or downturns, and are worth exploring further.

[3] In this case the pattern was a cycle – with appropriate interpolation, Fourier techniques would most certainly find it – but the approach here is completely general: any pattern can be used to form the dictionary.

[4] Finding the good set of patterns is referred to as **solving the inverse problem** in the paper.

[5] This is discussed in section 3.1 where the singular-value decomposition (SVD) is used to extract "eigenfaces". The eigenfaces have no simple form: They are certainly not cycles! Facial patterns emerge from the data itself, and a few (sparse) set of core patterns suffices for applications such as facial recognition.

Figure 2: Each trading day we fit 15 cycles to the stock price history from the previous 60 trading days, and used a 6 day prediction window. We require 3 or more cycles to have periods of more than 12 days (at least twice the prediction window) together with minima or maxima within the prediction window in order to signal that we should buy or sell respectively. This particular parameter combination avoided both 2000 and 2008 downturns experienced by the Nasdaq index. Running this strategy from 1992 through 2021, buying and selling full amounts, resulted in a final portfolio value of $1 957 642, corresponding to an annualized return rate of 13.40 % to the 11.25 % return of a single Nasdaq purchase in 1992 held until 2021.

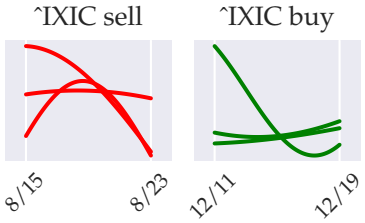


Figure 3: Here is a plot of the 3 cycles that achieved extremal values in the prediction window. **Left:** Three maxima between 8/15/2000 and 8/23/2000 result in a signal to sell the held shares. **Right:** Three minima between 12/11/2002 and 12/19/2002 result in a signal to buy as many shares as possible with the cash resulting from our previous sale.

More Details for the Technically Inclined

Fourier methods (cycles) are just one of an infinite family of orthogonal transformation methods and when we allow overcomplete, non-orthogonal representations, we can sometimes extract information much more effectively.

The core of this paper is an invitation to explore the information extraction power of the **sparse/low dimensional perspective** through **overcomplete dictionary methods** via the matching pursuit algorithm. By *sparse/low dimensional perspective* we refer (roughly) to the problem solving perspective that attempts to use relatively few basis elements (for example, pure Fourier components) to represent solutions or input data (or both). For example, more often than you might think, you might find that a great deal (for example, 99%) of the important information in a family of signals concentrated in a very small fraction (for example, 1%) of the possible signals. In fact, this is the basis of data compression – what we care about fills a tiny (small measure/low-dimensional) portion of the space of theoretically possible signals. To help make these methods instinctively available, we include critical pieces of the context from which dictionary methods emerged. These pieces argue that:

Sparse representations work and this is supported with several examples and the beginning of the discussion of the philosophy behind them.

The inverse problem perspective is the right perspective for understanding the implicit and explicit assumptions that effect what information we see and what we information we ignore. (An *inverse problem* uses *measurements* to infer the true state of the the process of thing we were measured. **Why this is often hard**: the measurement process is often complicated (hard to invert!) and lossy/noisy – you apparently lose information when you measure. **Example**: from a few, noisy, low resolution images of an evolving scene, find the true three dimensional, dynamic reality those images were measuring.)

Understanding the relation between metrics and prior assumptions is crucial if the nuances of the information extraction process are to be understood. In turn, these nuances are key to any sort of optimality that the analyst hopes to achieve when finding needles in haystacks.

Our GitLab Repository

To assist the more adventurous analysts in barehanded explorations, we provide code (and papers) in a GitLab repository (Link).