Predicting chaotic network time series
with a partial model

Franz Hamilton, 1 Tyrus Berry, 2 and Timothy Sauer 1,*

1 George Mason University, Fairfax, VA 22030
2 Pennsylvania State University, University Park, PA 16802

Methods for forecasting time series are a critical aspect of the understanding and control of complex networks. When the model of the network is unknown, nonparametric methods for prediction have been developed, based on concepts of attractor reconstruction pioneered by Takens and others. In this article we consider how to make use of a subset of the system equations, if they are known, to improve the predictive capability of forecasting methods. A counter-intuitive implication of the results is that knowledge of the evolution equation of even one variable, if known, can improve forecasting of all variables. The new method is illustrated on data from the Lorenz attractor and from a small network with chaotic dynamics.

PACS numbers: 87.19.lt, 05.45.Tp, 05.45.Jn, 87.19.li

One of the hallmarks of chaotic systems is the difficulty of accurate prediction. Due to exponential divergence of trajectories, long-term forecasting is seldom possible. On the other hand, short-term prediction is often feasible and significant progress has been achieved. In particular, when the system is known through time series observations alone, nonparametric methods have been developed to forecast chaotic trajectories.

For relatively low-dimensional chaotic dynamics, Takens’ method of attractor reconstruction [1–4] has long been the foundation of nonparametric time series prediction methods. Under suitable genericity hypotheses, the attractor may be represented by delay coordinate vectors built from the time series observations, and methods of prediction, control, and other applications from chaotic time series have been developed [5–7]. In particular, time series prediction algorithms locate the current position in the delay coordinate representation and use analogues from previously recorded data to establish a local, low-order predictive statistical model, which can be accomplished in several ways [8–21].

At the other end of the spectrum are parametric forecasting methods, applicable when a physically-motivated, complete model for the network is available. Nonlinear approaches to filtering [22–25] allow forecasting models to use the model equations to develop close to optimal predictions. Even if some variables are not observable, they may be reconstructed, provided that their model equations are known. However, when the model is not completely known, there has been little progress, and few methods are known that are able to span the gap between nonparametric and data assimilation methods.

In this Letter we consider the problem where not only is time series data available from a dynamical network, but in addition we possess a subset of the dynamical equations for the network. This situation reflects the case of a “partial model”, in which evolution equations for some, but not all, of the measured variables are available. We address the question of how best to extend nonparametric methods with this added, partial information for system prediction purposes.

This problem is endemic throughout the study of physical processes, where the amount of accessible data can easily exceed the availability of physical parametric models. In geophysical processes, basic principles may constrain a variable in terms of other driving variables in a way that is well understood, but the driving variables may be unmodeled or modeled with large error [26–29]. In a numerical weather prediction code, the physics may be known on the small scale but the large scale might be poorly modeled [30, 31]. In a fast-slow system governing excitable media, the slow variables are often driven in a known way by fast variables that are unmodeled [32, 33].

A specific example serves to illustrate the problem. Assume we can observe the $x, y$, and $z$ variables of the Lorenz-63 system [34]

$$\dot{x} = \sigma (y - x)$$
$$\dot{y} = x (\rho - z) - y$$
$$\dot{z} = xy - \beta z$$

where $\sigma = 10$, $\rho = 28$, $\beta = 8/3$, but that we have no knowledge of the generating equations. A reasonably successful nonparametric forecasting method with prediction horizon $T$ can be derived from attractor reconstruction techniques, using delay-coordinate versions of the current $x, y$, and $z$ to predict future $y$ values, for example. The direct prediction method begins by locating neighbors $[y(t'), y(t' - h), \ldots, y(t' - dh)]$ of the current delay coordinates $[y(t), y(t - h), \ldots, y(t - dh)]$ in the observed data, where $h$ is the sampling step size. The neighbors can be found by minimizing the Euclidean distance norm or any other appropriate norm. Then, the known values $y(t' + T)$ are used in a regression with a local model (typically locally-constant or locally-linear) to predict the future value $y(t + T)$. Since we know the $x$ and $z$ variables as well, we can include their delay co-

*Electronic address: tsauer@gmu.edu
coordinates to improve location of appropriate neighbors, which typically enhances the accuracy of the prediction of \( y(t + T) \).

Now assume that in addition to the time series data, we have extra knowledge in the form of a differential equation for one of the variables, say the \( y \) variable of the Lorenz system. In the current example, this assumption consists of knowing the single equation \( \dot{y} = x(\rho - z) - y \). Here we consider \( y \) to be the modeled variable, whose evolution is known in terms of the unmodeled variables \( x \) and \( z \); the evolution equations for \( x \) and \( z \) are considered to be unknown to us. The central question of this article is how to use the new information. Our proposed strategy is to use the differential equation to interpolate the known data, use the results to supplement the known data, and thereby substantially improve the accuracy of direct prediction methods.

![Graphs](image)

**FIG. 1:** Using one equation from the Lorenz system, the dataset is interpolated to reconstruct all of the system variables at subsample intervals. (a) Measured data \( y \)-values (blue circles) are interpolated by the multistep equation solver (solid/red trace), approximately matching the exact solution of the Lorenz equations (dashed trace). At each interpolation step, we use delay coordinates of the \( y \) variable to reconstruct the corresponding (unmodeled) (b) \( x \) and (c) \( z \) variables.

This is achieved by solving the differential equation on a finer sampling scale. Assume the data is obtained by sampling the system with step size \( h \). Subdivide the sampling interval into \( m \) smaller steps of length \( k \), or \( h = mk \). We will use the training data defined on the grid of step size \( h \), and the single differential equation for \( y \), to generate consistent \( x, y, z \) on the finer grid of step size \( k \). The procedure works as follows: At a given sampling time \( t_i \), apply a multistep quadrature method that uses previous \( x, y \), and \( z \) time series values on the step size \( h \) grid, and the differential equation for \( \dot{y} \), to approximate \( y(t + k) \), where \( k \) is the new, smaller step size. This will require a quadrature method for which the input and output step sizes can be arbitrarily adjusted, since the input data has step size \( h \) and the output has step size \( k \). We used a modified “fractional-step” multistep method that uses arbitrary \( h, k \) and can be built with arbitrary order. As an example, if \( h = 5k \), a useful fourth-order multistep method for \( \dot{y} = f(y) \) is

\[
y(t_i + k) \approx y(t_i) + k \sum_{j=1}^{4} b_j f(y(t_i - (j - 1)h))
\]

where \([b_1, b_2, b_3, b_4] = [3591, -1003, 533, -121]/3000.0 \]. We use an explicit method like (2) as a predictor, and then use an analogous implicit method as corrector to improve fidelity. The derivation and directions for carrying out an explicit or implicit fractional-step multistep quadrature method for arbitrary order and step sizes \( h \) and \( k \) are given in the Supplementary Material.

Once \( y(t_i + k) \) has been calculated on the entire time grid, we use the delay-coordinates of the \( y \) variable at the \( t_i + k \) times to reconstruct \( x(t_i + k) \) and \( z(t_i + k) \). Note the curious fact that although we only know the differential equation for \( y \), the modeled variable, we have used the single equation to bootstrap our knowledge to the unmodeled variables \( x \) and \( z \) at the new fine grid points. Now that we have computed values of \( x, y \) and \( z \) at all \( t_i + k \) times, we are in position to approximate \( y(t_i + 2k) \) with the same quadrature formula.

![Graph](image)

**FIG. 2:** Forecasting an individual trajectory of the Lorenz-63 \( y \) variable. Using the equation to supplement the forecast (red/solid curve) is more accurate than the direct forecast method alone (green squares). The dashed curve is the true \( y \) trajectory and the blue dots on the dashed curve denote the sampling rate (\( h = 0.2 \)).

The same idea is repeated to compute values at \( y(t_i + 2k) \), \( y(t_i + 3k) \), and so forth until all variables \( x, y \) and \( z \) are known on the finer \( k \)-grid. The result of this procedure is essentially an interpolation of the training set data, where the accuracy is leveraged by knowledge of the partial model. In Figure 1(a), we have available a training set of 2000 observations each of \( x, y \) and \( z \) from the Lorenz system at \( h = 0.2 \) time intervals. These intervals
are divided into \( m = 10 \) substeps, and a 4-step, fourth-order multistep quadrature method is used to integrate the known \( y \) equation. As discussed above, we can infer the \( x \) and \( z \) variables as well at the small stepsize, which are shown in Fig. 1(b-c).

Although we have described the method as working forward in time in steps of size \( k \) from the known data point on the \( h \)-grid, we can repeat the process backward in time and average the two results to improve fidelity of the reconstruction.

Fig. 3 compares mean forecasting error for the partial model method and the nonparametric method. The partial model method consistently outperforms the nonparametric method at all short-term forecasting horizons. Note a side benefit of the method: Even though the sampling rate of the data is sparse, predictions can be made not only at the original sampling rate but at any desired intermediate time horizon.

Now that we have shown a simple example in detail, we describe the method in more general terms. In the general network setting, assume \( x_1, \ldots, x_p \) are generic observable nodes in the sense of Takens, and nodes \( x_1, x_2, \ldots, x_r \), where \( r > p \), form the complete input set to \( x_1, \ldots, x_p \). Assume further that \( x_1, \ldots, x_r \) are observed in the training data at step size \( h \) and that we know the evolution equation of \( x_i \) for \( 1 \leq i \leq p \), say \( \dot{x}_i = f_i(x_1, \ldots, x_r) \). As above, we apply a multistep method to the known equations to bootstrap the values of \( x_1, \ldots, x_p \), and use a nonparametric method with the delay vectors of \( x_1 \) to approximate the remaining unmodeled variables at the same time points, resulting in an interpolation of all \( r \) variables at the smaller time step \( k \). This augmented training set is then used along with a nonparametric prediction algorithm to obtain forecasts for all \( r \) variables.

After supplementing the training data set with the new interpolated values, direct nonparametric prediction algorithms can be applied with the augmented data set. The combination of parametric and nonparametric methods results in a hybrid method which improves the predictability of the system compared with nonparametric methods alone. Fig. 2 shows an example of this improvement. Time series data of length 2000 is known up to time \( t = 0 \); the solid curve is the prediction going forward by the partial model method, compared with the exact trajectory (dashed curve). The circles represent the time series sampling rate from the known data, and the squares denote the nonparametric (direct forecasting) predictions without use of the partial model.

FIG. 3: Forecasting error statistics for the Lorenz-63 (a) \( y \), (b) \( x \) and (c) \( z \) trajectories over 5000 realizations. The data set consists of 2000 points sampled at \( h = 0.2 \). Using our knowledge of the \( y \) equation, we can interpolate the training data at subsample step size \( k = 0.02 \) and supplement the original training set. Direct forecasting using this interpolated data set (lower trace) outperforms direct forecasting with access to only the original data set (upper trace). Note the finer forecasting resolution of the lower trace. (d) Even under a moderate amount of observational noise (8% of the standard deviation), our hybrid forecasting offers better short-term prediction of the \( y \)-variable than the nonparametric forecast.

FIG. 4: Improved short-term forecasting of a small network. (a) Example interpolation of one node from the four node chaotic network driven by Lorenz-63. The dataset (blue circles) is used to interpolate Lorenz-96 (red curve). (b) At each step of the interpolation, delay-coordinates reconstruct the unmodeled Lorenz-63 forcing term (red curve). (c) Mean forecasting error of the chaotic network over 300 realizations. Direct forecasting using this interpolated dataset (lower trace) outperforms direct forecasting with access to only the original dataset (upper trace).

As an illustrative example we consider a network of
$p + 1$ nodes, comprised of a central Lorenz-63 attractor driving a coupled ring of $p$ Lorenz-96 nodes [35]. We assume that we know the differential equation of the ring nodes

$$\dot{x}_i = (x_{i+1} - x_{i-2})x_{i-1} - c_i x_i + F_i + b_i x_{p+1}^4$$  \hspace{1cm} (3)

for $i = 1, \ldots, p$, and lack knowledge of the evolution equations for all three Lorenz-63 variables $x_{p+1}^{(1,2,3)}$. In other words, we are driving the Lorenz-96 ring with the $x$-coordinate of the classical Lorenz attractor, while assuming that we know the Lorenz-63 equations but not the classical Lorenz equations. We form the training set from observations of variables $x_1, \ldots, x_p, x_{p+1}^1$ (not including the $y$ and $z$ variables of the classical Lorenz, since these variables do not occur in (3)) and attempt to forecast the Lorenz-96 system.

Fig. 4 shows results from a Lorenz-96 ring consisting of $p = 4$ nodes with $F_i = 2$ and $b_i = 1/8$ for $i = 1, \ldots, 4$ and $[c_1, c_2, c_3, c_4] = [1, 1.2, 1.4, 1.6]$. We collect 4000 data points from each node of the network, sampled at $h = 0.4$. Without knowledge of equation (3), we can use nonparametric methods to forecast (upper trace in Fig. 4(c)). Assuming knowledge of equation (3), we can interpolate the training data from each node (Fig. 4(a)) and simultaneously reconstruct the Lorenz-63 forcing term (Fig. 4(b)) at step size $k = 0.05$. By augmenting the training data with these interpolated values, the hybrid prediction (lower trace in Fig. 4(c)) offers an improvement in short-term prediction accuracy compared to the nonparametric method.

As a hybrid method, this forecasting approach attempts to explore the middle ground between data-driven statistics, where no model information is known, and the parametric case with complete knowledge of the model. We have tried to place the idea in as general a setting as possible, but it is clear that many further adaptations can be formulated, depending on the different sampling rates of distinct variables, and varying levels of confidence in the separate model equations. The fractional-step integrators discussed here are the foundation of the idea, and generalizations to partial differential equation models may also be feasible and effective.

The limitations of this new hybrid prediction method are similar to those faced by nonparametric approaches: As the effective dimension of the dynamics increases, even short-term prediction becomes more difficult. The role of noise is also important. We have emphasized the noiseless case to more clearly explain the method. For data with observational or dynamical noise, the partial model assumption that a differential equation is satisfied no longer holds exactly. In such cases the method should be generalized to include noise in the underlying model. An optimal approach for this case is left for future work.

**Acknowledgments.** The research was partially supported by grants CMMI-1300007, DMS-1216568 and DMS-1250936 from the National Science Foundation.


Supplementary Material.

Fractional-step multistep methods.

The general formula for the $s$-step modified multistep methods we use in the main text is

$$w_{i+k/h} = w_i + k[b_0 f_{i+k/h} + b_1 f_i + b_2 f_{i-1} + \ldots + b_s f_{i-s+1}]$$

Here $w_i$ denotes the approximate solution, $y' = f(t, y)$ is the differential equation, the step size is $h = t_i - t_{i-1} = t_{i-1} - t_{i-2} = \ldots$, and $f_i \equiv f(t_i, w_i)$. If $k = h$, this corresponds to the classical Adams-Bashforth-Moulton methods.

To derive the coefficients, we assume the previous steps are exact: $w_i = y_i = y(t_i)$ and $f_i = f(t_i, y_i)$. We want to approximate the solution $y_{i+k/h} = y(t_i+k)$, where $k$ may be different from $h$. Expanding with Taylor polynomials,

$$w_{i+k/h} = w_i + k[b_0 f_{i+k/h} + b_1 f_i + b_2 f_{i-1} + \ldots + b_s f_{i-s+1}]$$

must match the exact value

$$y_{i+k/h} = y_i + \frac{k}{1!} y_i' + \frac{k^2}{2!} y_i'' + \ldots + \frac{k^s}{s!} y_i^{(s)} + O(k^{s+1})$$

Setting $b_0 = 0$ to make the method explicit. Then the $y_i'$ terms imply $b_1 + b_2 + \ldots + b_s = 1$. For $1 \leq j \leq s - 1$, matching the $y_i^{(j)}$ terms yields

$$\frac{kb_2(-h)^j}{j!} + \frac{kb_3(-2h)^j}{j!} + \ldots + \frac{kb_s(-(s-1)h)^j}{j!} = \frac{k^{j+1}}{(j+1)!}$$

or setting $C = -h/k$,

$$C^j b_2 + (2C)^j b_3 + \ldots + ((s-1)C)^j b_s = \frac{1}{j+1}.$$  

Matching terms results in $s$ linear equations in $b_1, \ldots, b_s$, leaving a remainder of terms proportional to $k^{s+1}$ and $(|C|k)^{s+1}$, which become the truncation error. For $|C|$ on the order of 1, the $s$-step explicit multistep method is an order-$s$ solver. For large $|C|$, order considerations may be more complicated.

We can solve for $b_1, \ldots, b_s$ by solving the system

$$\begin{bmatrix} C & 2C & 3C & \ldots & (s-1)C \\ C^2 & (2C)^2 & (3C)^2 & \ldots & ((s-1)C)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C^{s-1} & (2C)^{s-1} & (3C)^{s-1} & \ldots & ((s-1)C)^{s-1} \end{bmatrix} \begin{bmatrix} b_2 \\ b_3 \\ \vdots \\ b_s \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/3 \\ \vdots \\ 1/s \end{bmatrix}$$

and then setting $b_1 = 1 - b_2 - \ldots - b_s$.

We recover the traditional coefficients by setting $k = h$. For example, the Adams-Bashforth three-step method requires solving

$$\begin{bmatrix} -1 & -2 & 1 \\ 1 & 4 & -3 \\ 0 & 16 & -51 \end{bmatrix} \begin{bmatrix} b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/3 \\ 1/6 \end{bmatrix},$$

which yields $b_1 = 23/12, b_2 = -4/3$, and $b_3 = 5/12$.

To advance the equation a half-step, or $k = h/2$, the coefficients are quite different. For the modified three-step explicit method, the coefficients satisfy

$$\begin{bmatrix} -2 & -4 & 16 \\ 4 & 16 & -71 \end{bmatrix} \begin{bmatrix} b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/3 \end{bmatrix},$$

implying $b_1 = 17/12, b_2 = -7/12$, and $b_3 = 1/6$.

The derivation of the implicit formulas are similar except for allowing arbitrary $b_0$. In this case the equations are

$$\begin{bmatrix} 1 & C & 2C & \ldots & (s-1)C \\ C^2 & (2C)^2 & (3C)^2 & \ldots & ((s-1)C)^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C^{s} & (2C)^{s} & (3C)^{s} & \ldots & ((s-1)C)^{s} \end{bmatrix} \begin{bmatrix} b_0 \\ b_2 \\ \vdots \\ b_s \end{bmatrix} = \begin{bmatrix} 1/2 \\ 1/3 \\ \vdots \\ 1/(s+1) \end{bmatrix}$$

followed by setting $b_1 = 1 - b_0 - b_2 - \ldots - b_s$. Note that the unmatched terms are proportional to $k^{s+2}$, so the implicit $s$-step method is an order $s + 1$ solver for reasonably small $|C|$.  