Program in Applied Mathematics

Data-driven Model Reduction by Wiener Projection

Jared McBride

Applied Mathematics
University of Arizona

Sept 18, 2020

THE UNIVERSITY OF ARIZONA

Outline

Program in Applied Mathematics

The Problem

Program in Applied Mathematics

Many important models today contemplate

- large number of degrees of freedom
- across many orders of magnitude in space and time, without sharp scale separation.

Examples: Power flow on large grids, neural activity in the brain, weather forecasting, etc.

Some tasks require repeated model runs such as for

- Uncertainty quantification
- Optimization and control

Commonly, only a relatively small number of variables are of direct interest or even observable.

Goal and Difficulties

The **goal** is then to find reduced order models, which include only the variables of interest (resolved variables), capable of finite time forecasting as well as reproducing long-time statistics like correlation functions and marginals of stationary distributions, at lower computational costs.

How can the effect of unresolved variables be approximated by using the resolved variables and stochastic terms.

Unlike under situations with sharp scale separation,

- memory (marginals of Markov process may not be Markov) and
- noise effects

must be accounted for in many applications.

Wiener Filter

Given two stationary processes $\mathbf{x}_n$,$\mathbf{y}_n$ The Wiener Filter computes a linear least square estimate $\hat{\mathbf{y}}_n$ of a process $\mathbf{y}_n$ given $\mathbf{x}_n$, for this reason

- $\mathbf{y}_n$ is called the signal,
- $\mathbf{x}_n$ are called the predictors.

This means we seek an h such that

$$\mathbb{E}\|\mathbf{y}_n - (\mathbf{x} \star h)_n\|^2 = \text{minimum}$$

In our case we want to require h_n to be

- causal (meaning $h_n = 0$ for $n < 0$)
- rapid decay (so that efficiency is gained)

Wiener Filter
Causal Solution

Basically give the spectral density of **yx**, $S_{\mathbf{x}}(z)$, and the cross spectral density of $\mathbf{y}$ and $\mathbf{x}$, $S_{\mathbf{yx}}(z)$ The causal solution is obtained by the formula below.

$$H(z) = \left\{ S_{\mathbf{yx}}(z) {S^{+}}_{\mathbf{x}}^{-1}(z) \right\}_{+} {S^{-}}_{\mathbf{x}}^{-1}(z)$$

Here S^{+} and S^{-} form a special factorization of S_x such that $S^{+}(z) = S^{-*}(z^{-*})$,

$$S(z) = S^{+}(z)S^{-}(z) \qquad \text{for } z \in \partial\mathbb{D}$$

and $S^{-}(z)$ is minimum phase, meaning all it's poles and zeros are strictly inside the unit circle. This makes $S^{-}(z)$ a causal and casually invertable LTI system.

Spectral Factorization

More specific

Program in Applied Mathematics

our favorite version of Spectral Factorization Theorem

If $\mathbf{y}$ is a mean zero, stationary, discrete time stochastic d-vector-valued process that admits a rational z-spectrum $S_{\mathbf{y}}$ analytic on some annulus containing the unit circle, and

$$S_{\mathbf{y}} > 0 \qquad \text{everywhere on } \partial\mathbb{D}.$$

Then there exists matrix functions $S^{+}(z)$ and $S^{-}(z)$, such

- $S^{+}(z)$ is a $d \times d$ rational matrix function that is analytic on and inside the unit circle,
- $S^{+-1}(z)$ is analytic on and inside the unit circle.
- $S^{-}(z) = S^{+*}(z^{-*})$ and
- $S(z) = S^{+}(z)S^{-}(z)$.

Spectral Factorization

More specific

Notice that the factorization if non unique in this form, since is $U(z)$ is unitary on $\partial\mathbb{D}$ and $S^{-}(z)$ is a spectral factor of $S(z)$ then so is $U(z)S^{-}(z)$.

It would be nice to leverage this to improve performance. (possible future work)

Spectral Factorization
Numerical

Most Numerical algorithms assume $S(z)$ is rational and has the form of a Laurent Polynomial meaning it may be written as

$$S(z) = \sum_{n=-m}^{m} c_n z^{-n} \qquad \text{with } c_n = c^*_{-n}.$$

If this is assumed it may be shown that

$$S^+(z) = \sum_{n=1} L_n z^n \qquad \text{and} \qquad S^-(z) = \sum_{n=1} L^*_n z^{-n}$$

Algorithms that use Toeplitz matrices.

- Schur
- Levinson-Durbin

Algorithms that use State Space formulations.

- Riccati Equation
- Kalman Filter
- Chadrasekhar-Kailath-Morf-Sidhu (CKMS)

This Approach

Given a full model

$$X_n = F(X_n)$$

with resolved variables collected in x_n, select functions $\psi^{(i)}(x)$ (informed by model) on reduced state variables (resolved variables).

$$\psi(x) = \left(\psi^{(0)}(x)\middle|\psi^{(1)}(x)\middle|\cdots\middle|\psi^{(\nu)}\right) \qquad \text{and} \qquad \psi_k = \psi(x_k).$$

ψ_k are our predictors, x_k is the signal, two stationary stochastic processes. Use WF to find LTI filter to approximate X_k by ψ_k optimally. This forms a "reduced" model, closed in the variables of interest.

$$\boxed{x_{n+1} = \sum_{k=0}^{\infty} \psi_k \cdot h_{n-k} + \xi_{n+1}}$$

We use the data to infer h_k (WF) and ξ_n.

This Approach

Why “reduced”?
If h_k doesn’t decay well, what do we gain?
Let $y_n = \sum_{k=0}^{\infty} \psi_k \cdot h_{n-k}$ be an auxiliary variable.
If we could take the z-transform of both sides we would get

$$Y(z) = \Psi(z) \cdot H(z)$$

further if had had a good rational approximation $H(z) = A(z)/B(z)$ we could write $Y(z)A(z) = \Psi(z) \cdot B(z)$ and

$$y_n + a_1 y_{n-1} + \cdots + a_p y_{n-p} = \psi_{n-p} \cdot b_0 + \psi_{n-p+1} \cdot b_1 + \ldots \psi_{n-p+q} \cdot b_q$$

where $A(z) = z^p + a_1 z^{p-1} + \cdots + a_{p-1} z^1$ and $B(z) = b_0 + b_1 z^1 + \cdots + b_q z^q$

How to solve

Dr. Kevin Lin and Dr. Fei Lu solves the "WF" (a and b) in time domain, using an numerical optimization optimization algorithm.

This study investigates computing the Wiener filter by spectral methods (that is, employing information like the power spectra S_{yx} and S_x). This is a direct method requiring no iterative optimization.

Advantages:

- Quicker
- more accurate (?)

WF Example: AR(2) Signal, Filtered, Ad. W

Let us consider again the stationary autoregressive process of order 2,

$$y_n = (r_1 + r_2)y_{n-1} - r_1 r_2 y_{n-2} + u_n, \qquad \text{for } n > -\infty$$

for $r_1, r_2 \in \{z : |z| < 1\}$. This time however, we define the observations to be the signal y operated upon by a finite impulse response time invariant filter w with additive white noise.

$$x_n = (w * y)_n + v_n, \qquad \text{for } n > -\infty.$$

For simplicity let

$$w = (\dots, 0, \boxed{1}, w_1, w_2, 0, \dots),$$

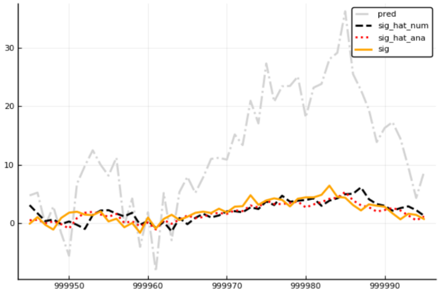
where the box indicate the element indexed by 0 and $w_1, w_2 \in \mathbb{R}$, then write

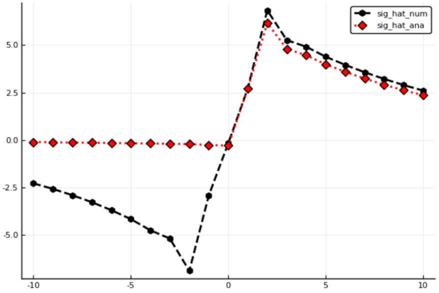
$$W(z) = \sum_{k=-\infty}^{\infty} w_k z^{-k} = 1 + w_1 z^{-1} + w_2 z^{-2}.$$

WF Example: AR(2) Signal, Filtered, Ad. W

Program in Applied Mathematics

Here is a run, with $r_1, r_2 = -0.2, 0.9$, $w_1, w_2 = -0.1, 5$, and $\sigma_{\mathbf{v}} = 1.1$. The trajectory has 10^6 steps after discarding 10^3 steps.



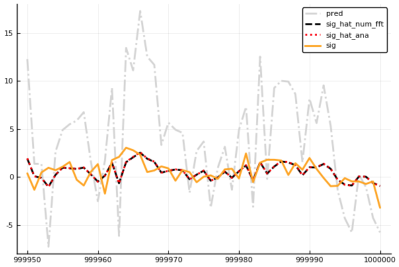


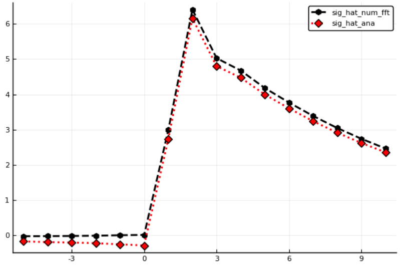
Left: A window of the time series for the signal (orange), the predictors (light gray), the estimated signal using the analytic and numerical Wiener filter (red, black).

Right: The covariance between errors (red from analytic filter, black from numerical) and predictors (observations).

WF Example: AR(2) Signal, Filtered, Ad. W

New results! The problem was in the function used to estimate the cross spectrum. It was a subtle bug I didn't find till I worked with complex-valued time series, in which case the bug was more apparent.





Left: A window of the time series for the signal (orange), the predictors (light gray), the estimated signal using the analytic and numerical Wiener filter (red, black).

Right: The covariance between errors (red from analytic filter, black from numerical) and predictors (observations).

MR Example: Nonlinear Langevin

Double-welled Potential

Program in Applied Mathematics

In this example we consider the stochastic, over-damped Langevin equation with a double-welled potential $V(x) = \frac{1}{4}(x^2 - 1)^2$. So, we consider the following SDE.

$$dX_t = -X_t(X_t^2 - 1)dt + dB_t$$

To find a numerical solution we first use the Euler-Maruyama scheme

$$X_{n+1} = X_n - hX_n(X_n^2 - 1) + e_n$$

here

$$e_n \sim (B_{t_{n+1}} - B_{t_n}) \sim N(0, h) \sim \sqrt{h}N(0, 1).$$

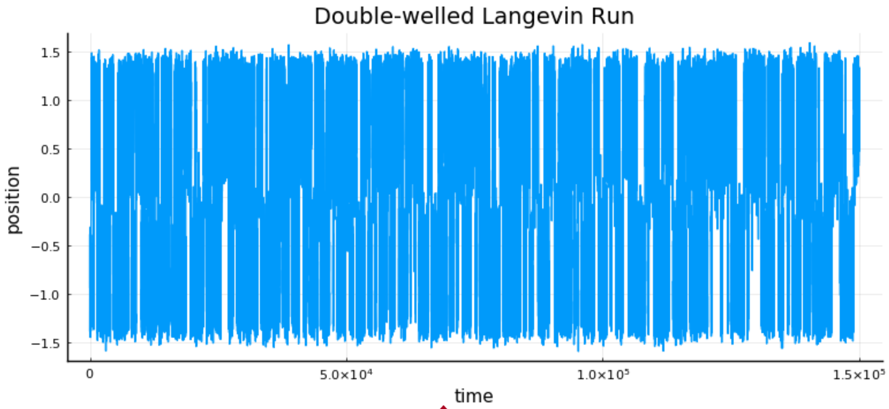
Now we generate the sample path.

MR Example: Nonlinear Langevin

Double-welled Potential

Program in Applied Mathematics

The full was generated by Euler-Maruyama 150 million ($1.5 \cdot 10^8$) steps, ($1.5 \cdot 10^5$ secs at a time step of 10^{-3}) $\sigma = .35$ we discarded 10^7
The observations seen below are every 100th point.



MR Example: Nonlinear Langevin
Double-welled Potential

Program in Applied Mathematics

Recall the original model was

$$X_{n+1} = X_n - hX_n(X_n^2 - 1) + e_n$$

so we will choose $\psi(x) = [x; x^3]$ We feed these into the Wiener filter function and get the WF.

Karumoto-Sivashinsky Equation

In one demnsional space is

$$u_t + uu_x + u_{xx} + u_{xxxx} = 0, \qquad t \in [0, \infty), x \in [o, L]$$

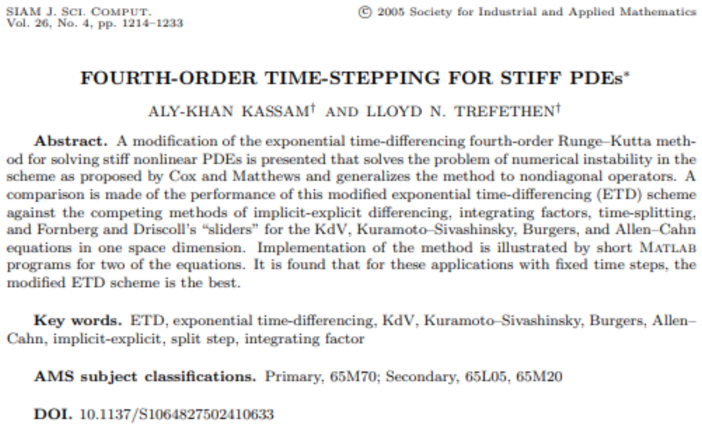
In Fourier coefficients this is

$$\dot{v}_k = -\frac{iq_k}{2} \sum_l v_l v_{k-l} + (q_k^2 - q_k^4) v_k, \qquad q_k = \frac{2\pi k}{L}$$

this is now an ODE in the Fourier modes, we now use a numerical scheme to solve this ODE.

Karumoto-Sivashinsky Equation

These ODE's tend to be stiff so we use an exponential time differencing (ETD) scheme to solve them. This scheme is paired up with an RK4 method.



SIAM J. SCI. COMPUT.
Vol. 26, No. 4, pp. 1214–1233

© 2005 Society for Industrial and Applied Mathematics

FOURTH-ORDER TIME-STEPPING FOR STIFF PDEs*

ALY-KHAN KASSAM[†] AND LLOYD N. TREFETHEN[†]

Abstract. A modification of the exponential time-differencing fourth-order Runge–Kutta method for solving stiff nonlinear PDEs is presented that solves the problem of numerical instability in the scheme as proposed by Cox and Matthews and generalizes the method to nondiagonal operators. A comparison is made of the performance of this modified exponential time-differencing (ETD) scheme against the competing methods of implicit-explicit differencing, integrating factors, time-splitting, and Fornberg and Driscoll's "sliders" for the KdV, Kuramoto–Sivashinsky, Burgers, and Allen–Cahn equations in one space dimension. Implementation of the method is illustrated by short MATLAB programs for two of the equations. It is found that for these applications with fixed time steps, the modified ETD scheme is the best.

Key words. ETD, exponential time-differencing, KdV, Kuramoto–Sivashinsky, Burgers, Allen–Cahn, implicit-explicit, split step, integrating factor

AMS subject classifications. Primary, 65M70; Secondary, 65L05, 65M20

DOI. 10.1137/S1064827502410633

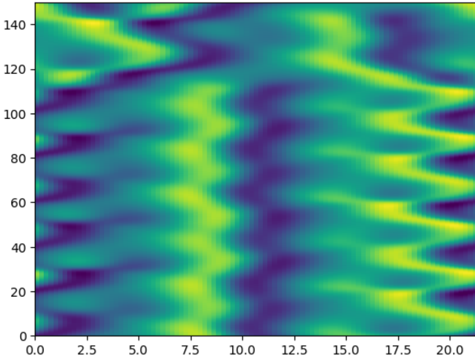
Karumoto-Sivashinsky Equation

This introduces the full model of the form

$$V_{n+1} = F(V_n)$$

Where F steps the vector $V_n = (v_k^n;\ n \in \mathbb{Z})$ forward in time. Generally, because of the decay of the Fourier modes we consider only the first so many modes, about 100.

Displayed used 96 modes, 10^8 I discarded half, The time step is 10^3.



Difficulties

The theoretically two ingredients are $S_{X\Psi}(z)$ and $S_X(z)$. Rather, $S_{X\Psi}(z)$ and $S_X^-(z)$ ($S_X^+(z)$ $S_X^-(z)$ derives from we need What we have is data. These can be difficult to estimate. Here is an illustration:

WF: AR(2) Signal, Filtered, Ad. WN

Estimating $S_{X\Psi}(z)$

Observe that

$$S_{\mathbf{yx}}(z) = S_{\mathbf{y}}(z)W^*(z^{-*}) \qquad \text{and} \qquad S_{\mathbf{x}} = W(z)S_{\mathbf{y}}(z)W^*(z^{-*}) + \sigma_{\mathbf{v}}^2.$$

So,

$$S_{\mathbf{yx}}(z) = \frac{1 + w_1 z + w_2 z^2}{(1 - r_1 z^{-1})(1 - r_2 z^{-1})(1 - r_1^* z)(1 - r_2^* z)}$$

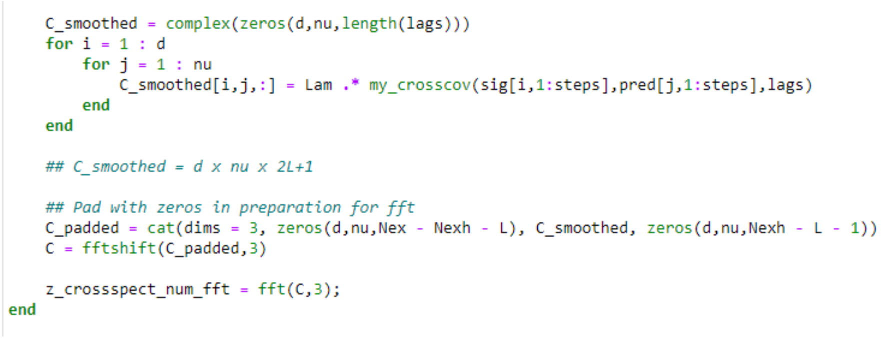
Numerically this is processed in

1. estimate cross covariances
2. smooth these
3. take z-transform (fft)

WF: AR(2) Signal, Filtered, Ad. WN

Estimating $S_{X\Psi}(z)$

Here's a snippet:

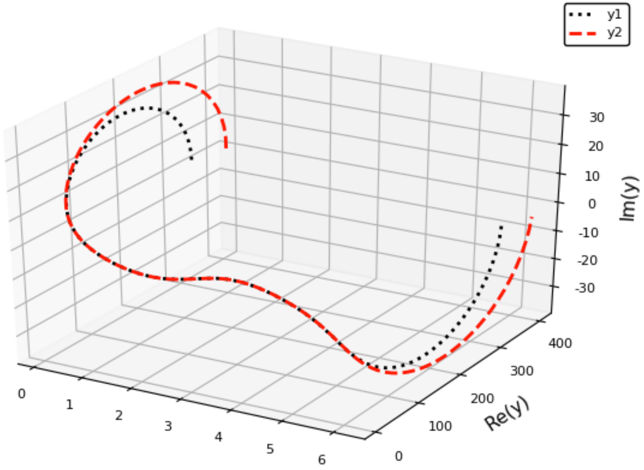


```
    C_smoothed = complex(zeros(d,nu,length(lags)))
    for i = 1 : d
        for j = 1 : nu
            C_smoothed[i,j,:] = Lam .* my_crosscov(sig[i,1:steps],pred[j,1:steps],lags)
        end
    end

    ## C_smoothed = d x nu x 2L+1

    ## Pad with zeros in preparation for fft
    C_padded = cat(dims = 3, zeros(d,nu,Nex - Nexh - L), C_smoothed, zeros(d,nu,Nexh - L - 1))
    C = fftshift(C_padded,3)

    z_crossspect_num_fft = fft(C,3);
end
```

WF: AR(2) Signal, Filtered, Ad. WN

Estimating $S_{X\Psi}(z)$

Here's the result:



WF: AR(2) Signal, Filtered, Ad. WN

Estimating $S_{\Psi}^{-}(z)$

Program in Applied Mathematics

For this, analytically we have we have

$$S_{\mathbf{x}}(z) = \frac{w_2 + \sigma_{\mathbf{v}}^2 r_1^* r_2^*}{\rho_1^* \rho_2^*} \cdot \frac{(1-\rho_1 z^{-1})(1-\rho_2 z^{-1})(1-\rho_1^* z)(1-\rho_2^* z)}{(1-r_1 z^{-1})(1-r_2 z^{-1})(1-r_1^* z)(1-r_2^* z)}$$

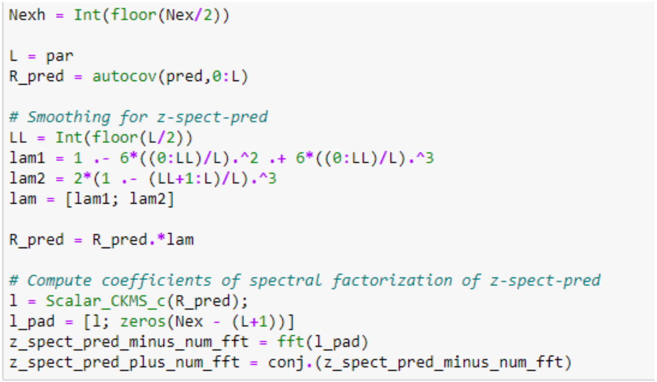
Numerically this is processed in

1. estimate autocovariance
2. smooth this
3. feed it into factorizing script
4. take fft

WF: AR(2) Signal, Filtered, Ad. WN

Estimating $S_{X\Psi}(z)$

Here's a snippet:

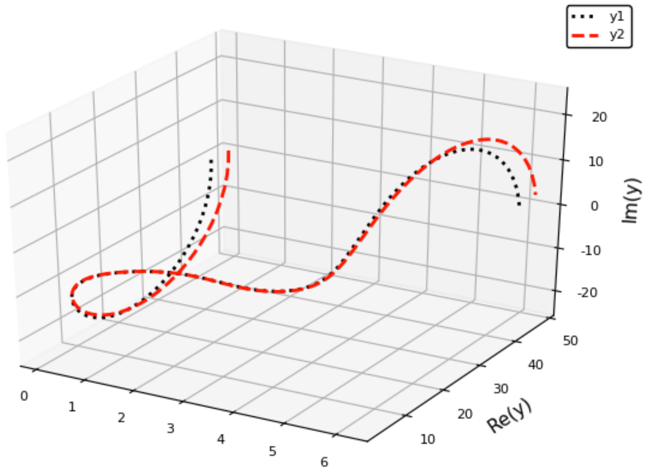


```
Nexh = Int(floor(Nex/2))

L = par
R_pred = autocov(pred,0:L)

# Smoothing for z-spect-pred
LL = Int(floor(L/2))
lam1 = 1 .- 6*((0:LL)/L).^2 .+ 6*((0:LL)/L).^3
lam2 = 2*(1 .- (LL+1:L)/L).^3
lam = [lam1; lam2]

R_pred = R_pred.*lam

# Compute coefficients of spectral factorization of z-spect-pred
l = Scalar_CKMS_c(R_pred);
l_pad = [l; zeros(Nex - (L+1))]
z_spect_pred_minus_num_fft = fft(l_pad)
z_spect_pred_plus_num_fft = conj.(z_spect_pred_minus_num_fft)
```

WF: AR(2) Signal, Filtered, Ad. WN

Estimating $S_{X\Psi}(z)$

Here's the result:



Thank you!