Finding Stationary States of Noisy Processes

C. W. Gear

October 3, 2002

Abstract

We discuss the problem of estimating the "stationary state" of an iterative process that is generating a sequence of vectors whose output is contaminated by noise in various ways. If the dimension of the system is not too large, it is also possible to estimate the eigenvalues of the system near equilibrium. The method can also be adapted to finding an underlying "smooth" function if the iterative process is, in fact, yielding a noisy approximation to a continuously changing function.

Keywords Stochastic Processes, Steady States

1 Introduction

We suppose we have an iterative process, such as a one-step integrator, that generates a sequence of M-dimensional vectors, $y_n, n = 0, \cdots$, according to the process

$$y_{n+1} = \Theta(y_n) + r_n \tag{1}$$

where r_n is the M-dimensional noise introduced by the process. (For example, the "integrator" could actually be a Monte Carlo simulation of an evolutionary process.)

We wish to compute a stationary state of Θ, that is, a z such that

$$z = \Theta(z)$$

The approach we will use will be to compute some maximum likelihood estimate based on some assumptions about the process and the noise. As far as the noise goes, we will assume that it consists of independent Gaussian noise with the same variance in each component. (Obviously one could also consider estimating the variance component by component if it was thought that this assumption was significantly in error.)

We will also assume that we are near the stationary state and that near this state it is reasonable to use a linear approximation to Θ. (This implicitly assumes that the noise is small enough compared to the second derivatives of Θ so that a linear approximation is reasonable.) Let the Jacobian of Θ be J. Then we have

$$\begin{aligned} y_{n+1} &= \Theta([y_n - z] + z)) + r_n & (2)\\ &= \Theta(z) + J[y_n - z] + r_n & (3)\\ &= z + J[y_n - z] + r_n & (4) \end{aligned}$$

Let

$$w = (I - J)z \tag{5}$$

Then we have

$$r_n = y_{n+1} - Jy_n - w$$

Under the assumption that r_n is independent Gaussian, we can simply compute J and w to minimize

$$S = \sum_{n=1}^{N-1} ||y_{n+1} - Jy_n - w||_2^2 \tag{6}$$

This is a linear least squares problem for J and w. Then we find the stationary state, z, by solving (5). Thus, a possible iterative process for computing a stationary state is the following:

1. Choose an initial estimate of z, say z_0 and set $j = 0$.
2. Set $y_0 = z_j$.
3. Use eq. (1) to compute N values of $y_1, \cdots, y_N$ where $N > M + 1$ (much greater if the noise is large).
4. Minimize S in eq, (6) for J and w.
5. Use eq. (5) and compute $z_{j+1} = (I - J)^{-1}w$.
6. Set $j \leftarrow j + 1$ and repeat steps 2 - 7 until convergence.

Once we have estimated J we can then compute its eigenvalues to estimate the eigenvalues of the process.

There are several potential problems with the method outlined above. The most obvious is that eq. (6) involves $M(M + 1)$ unknowns, where M is the dimension of the vector y. If M is large, it is not practical. The second is that if the estimate for the matrix J has an eigenvalue of one, eq. (5) is singular, and is ill-conditioned if an eigenvalue is close to one.

There are two approaches to the high dimensionality problem. The first is to use a suitable restriction of y to a lower dimensional space as was done, for eaxmple, in [2] and [1]. The least squares fit is then done in this lower-dimensional space to get a new estimate of the stationary state in the low dimensional space. This result has to be lifted back to the original space to get a starting value for the iteration (1). This approach requires that we can identify a low-dimensional space that will contain a reasonable approximation to the sequence of y_n's.

A second approach is to assume that there is only one "slow" eigenvalue in the process, so to assume that J can be taken as a scalar. In that case, there are only $M + 1$ unknowns to find in the least squared process, so a much smaller N can be used in step 3 above.

The near singularity of $J - I$ presents difficulties. If $J - I$ is truly singular, it means that $z = \Theta(z)$ does not have a unique solution locally. If we had computed a z_j that is a solution, if the noise in subsequent iterations causes us to take a random walk in the manifold of local solutions with small perturnations off the manifold, and if it can be approximated as a linear manifold, the best estimate we could make is simply the least

squared fit of z to the set of iterates, y_i, in other words, the average. This suggests an approach: use a suitable combination of the average of the y_i and the solution of eq. (5) to estimate z where the combination is chosen so the if $J-I$ is far from singular, eq. (5) is favored, whereas the average is favored as $J-I$ approaches singularity. This can be done with a "penalty function" type approach in which we compute z as the minimizer of

$$||(J-I)z - w||_2^2 + \mu||z - \sum_{i=0}^{N} y_i/(N+1)||_2^2 \tag{7}$$

where μ is a small positive value (dependent on M and machine precision).

2 An Extension to Smooth Functions

The above method can be viewed as an attempt to compute a constant function, z, as an approximation to the solution of the iterative process. If we are performing an integration, rather than finding a stationary state, we might want to find a smooth funtion, $z(t)$, that approximates the values found from the iteration. The above process can easily be extended as follows.

Suppose that we want to approximate each of the components of the solution by a low degree polynomial (any other approximation could be used).

Now we assume that our underlying process in the absence of noise gives

$$z(t_{n+1}) = \Theta(z(t_n)) \tag{8}$$

and, in the presence of noise we rewrite eq. (1) as

$$\begin{aligned} y_{n+1} &= \Theta([y_n - z(t_n)] + z(t_n)) + r_n & (9)\\ &= \Theta(z(t_n)) + J[y_n - z(t_n)] + r_n & (10)\\ &= z(t_{n+1}) + J[y_n - z(t_n)] + r_n & (11) \end{aligned}$$

Now let

$$w_n = z(t_{n+1}) - Jz(t_n)$$

so we have

$$y_{n+1} = w_n + Jy_n + r_n$$

Let the approximation to $z(t)$ be expressed in terms of the m basis functions $\phi_k(t), k = 1, \cdots, m$. Then we want to find a $z(t)$ given by

$$z(t) = \sum_{k=1}^{m} \alpha_k \phi_k(t) \tag{12}$$

(Note that the m α_k are each column vectors of dimension M.) Let $\Phi(t)$ be a column vector whose m entries are $\phi_k(t)$. Since the $\phi_k(t)$ are a basis set (for polynomials or any other m-dimensional approximation), there exists a "forward step" operator, E, such that

$$\Phi(t_{n+1}) = E\Phi(t_n)$$

Let A be the M by m matrix whose columns are $\alpha_k, k = 1, 2, \cdots, m$, and let Φ_p be the m by N matrix whose n-th column is $\Phi(t_n), n = p, \cdots, N + p - 1$, where $p = 0$ or 1. Thus

$$\Phi_1 = E\Phi_0$$

If we define

$$Z_0 = A\Phi_0 \tag{13}$$

and

$$Z_1 = A\Phi_1 = AE\Phi_0 \tag{14}$$

then the columns of Z_p are z_n for $n = p, \cdots, N + p - 1$. Define, similarly, the matrices Y_p from the computed values, y_n, and R_0 from the noise vectors, r_n.

Defining B by

$$B = AE - JA \tag{15}$$

we can write eq. (11) as

$$\begin{aligned} Y_1 &= Z_1 - JZ_0 + JY_0 + R_0 & (16)\\ &= B\Phi_0 + JY_0 + R_0 & (17) \end{aligned}$$

This gives a linear least squares problem for B and J which can be written as

$$\min_{J,B} \text{trace}([Y_1 - JY_0 - B\Phi_0][Y_1 - JY_0 - B\Phi_0]^T) \tag{18}$$

Having found B and J we can solve for A using eq. (15). As before, we will run into problems if J has an eigenvalue near 1 because E has an eigenvalue equal to one. As before we can instead minimize a problem with a "penalty function," namely

$$\min_A [||(AE - JA)\Phi_0 - B|| + \mu||Y_1 - A\Phi_1||] \tag{19}$$

where the norm is the Frobenius norm (sum of squares).

3 Implementation

The solution of eq. (18) is straightforward. Differentiating w.r.t. J and B we get the linear equations

$$\begin{aligned} Y_0Y_0^TJ^T + Y_0\Phi_0^TB^T &= Y_0Y_1^T & (20)\\ \Phi_0Y_0^TJ^T + \Phi_0\Phi_0^TB^T &= \Phi_0Y_1^T & (21) \end{aligned}$$

Eq. (19) involves the minization of a quadratic function of A so yields a linear equation for A. However, A is "buried" inside so it is a little tedious wo write out for computer solution.

We can proceed by "reshaping" A into a vector (**reshape** is a MatLab function) as follows. Define the column vector $r(A)$ consisting of concatenation of the transposes of the rows of A in turn. In other words, if A is an n by m matrix:

$$r(A) = [A_{11}A_{12}\cdots A_{1m}A_{21}A_{22}\cdots A_{2m}\cdots A_{nm}]^T$$

Also define $c(A)$ to be the vector consisting of the columns of A, or

$$c(A) = r(A^T)$$

Defining the outer product of two matrices P and Q to be

$$P \otimes Q = \begin{bmatrix} P_{11}Q & P_{12}Q & \cdots & P_{1m}Q \\ P_{21}Q & P_{22}Q & \cdots & P_{2m}Q \\ & \vdots & & \\ P_{n1}Q & P_{n2}Q & \cdots & P_{nm}Q \end{bmatrix}$$

we find the following relations for the operators c and r on the matrix product UV where U is m by n and V is n by p:

$$r(UV) = (I_m \otimes V^T)r(U) = (V^T \otimes I_m)c(U) \quad (22)$$
$$r(UV) = (U \otimes I_p)r(V) = (I_p \otimes U)c(V) \quad (23)$$

We can rewrite the minimization problem (19) as

$$\min_A [v_1^T v_1 + \mu v_2^T v_2] \quad (24)$$

where

$$v_1 = r((AE - JA)\Phi_0 - B) \quad (25)$$
$$v_2 = r(A\Phi_1 - Y_1) \quad (26)$$

Using the relations above, we can express these as matrices times $r(A)$, namely

$$v_1 = (I_M \otimes \Phi_0^T)r(AE - JA) - r(B) \quad (27)$$
$$= (I_M \otimes \Phi_0^T)[(I_M \otimes E^T) - (J \otimes I_m)]r(A) - r(B) \quad (28)$$
$$= [(I_M \otimes \Phi_1^T) - (J \otimes \Phi_0^T)]r(A) - r(B) \quad (29)$$
$$v_2 = (I_M \otimes \Phi_1^T)r(A) - r(Y_1) \quad (30)$$

Then we can differentiate the expression in eq. (24) to get

$$[(I_M \otimes \Phi_1^T) - (J \otimes \Phi_0^T)]^T[(I_M \otimes \Phi_1^T) - (J \otimes \Phi_0^T) + \mu(I_M \otimes \Phi_1^T)^T(I_M \otimes \Phi_1^T)]r(A)$$
$$= [(I_M \otimes \Phi_1^T) - (J \otimes \Phi_0^T)]^T r(B) + \mu(I_M \otimes \Phi_1^T)^T r(Y_1)$$

which can be solved for $r(A)$.

References

[1] Gear, C. W., Kevrekidis, I. G., and Theodoropoulos, C., "Coarse" Integration/Bifurcation Analysis via Microscopic Simulators: micro-Galerkin Methods, NEC Research Institute Report 2001-106, to appear, Computers and Chemical Engineering.

[2] Projective Integration Methods for Distributions, NEC TR 2001-130, 26 Nov, 2001