Lecture 4
Lotka-Volterra Equation and Singular Values

Orthogonal Polynomials, Moments, Determinants, and Measure Deformation

Moody T. Chu

North Carolina State University

June 17, 2009 @ MCM, CAS

Outline

Outline

Outline

Singular Value Decomposition

SVD Dynamics

SVD Flow

Lotka-Volterra Equation

Integrable Discretization

Symplectic Euler Scheme

Equivalence to the Oldies

New Integrals

Conclusion

Outline

Singular Value Decomposition

SVD Dynamics

- SVD Flow
- Lotka-Volterra Equation

Integrable Discretization

- Symplectic Euler Scheme
- Equivalence to the Oldies
- New Integrals

Conclusion

SVD

- Given $A \in \mathbb{R}^{m \times n}$, there exist real orthogonal matrices $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ such that

$$A = U \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} V^\top$$

  where

$$\Sigma = \mathrm{diag}\{\sigma_1, \ldots, \sigma_r\},$$

  with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$.
- Critical decomposition with many important applications.

SVD Algorithm
(Golub & Kahan, '65)

- First reduce A to a bidiagonal matrix B_0 via orthogonal equivalence transformations.
- Critical components:
  - Performing the QR algorithm on the product $B_0^\top B_0$ without explicitly forming the product.
  - The bidiagonal structure is preserved throughout the iteration.

Make It Continuous

- Assume

$$B(t) = U(t)B_0V(t), \quad U(t) \in \mathcal{O}(m), V(t) \in \mathcal{O}(n).$$

- Necessary format:

$$\frac{dB}{dt} = BR - LB, \quad B(0) = B_0.$$

  - Coordinate transformation:

$$\begin{cases} \frac{dU}{dt} = -LU, \\ \frac{dV}{dt} = VR, \end{cases} \quad L, R \in o(n).$$

- The choice of skew-symmetric matrix parameters $L(t)$ and $R(t)$ determines the dynamics.

Maintain the Bidiagonal Structure

- Want
  - $B(t)$ remains bidiagonal for all t.
  - $L(t)$, $R(t)$ are tridiagonal and skew-symmetric.
  - Good convergence.
- Among many other choices,

$$
\begin{aligned}
L &= \Pi_0(BB^\top), \\
R &= \Pi_0(B^\top B).
\end{aligned}
$$

SVD Flow
(Chu, '86)

- Given a bidiagonal matrix B_0,

$$\frac{dB}{dt} = B\Pi_0(B^\top B) - \Pi_0(BB^\top)B, \quad B(0) = B_0,$$

  - $B(t)$ stays bidiagonal for all t.

Component Form

- Denote

$$B(t) := \operatorname{diag}\left\{ \begin{matrix} & b_2(t) & & \cdots & & b_{2n-2}(t) & \\ b_1(t) & & b_3(t) & & \cdots & & b_{2n-1}(t) \end{matrix} \right\},$$

- SVD flow in component form:

$$\begin{cases} \frac{db_{2k-1}}{dt} = b_{2k-1}(b_{2k}^2 - b_{2k-2}^2) & \text{for } 1 \le k \le n, \\ \frac{db_{2k}}{dt} = b_{2k}(b_{2k+1}^2 - b_{2k-1}^2) & \text{for } 1 \le k \le n-1. \end{cases}$$

  - $b_0 = b_{2n} \equiv 0$.

Asymptotic Behavior

- Define $Y(t) = B^\top(t)B(t)$. Then

$$\frac{dY}{dt} = [Y, \Pi_0(Y)].$$

  - Convergence follows from the Toda dynamics.

- The sequence $\{B(\ell)\}$ by sampling $B(t)$ at integer times corresponds to the iterates produced by the Golub-Kahan SVD algorithm.

Lotka-Volterra Equation

- Change variables,

$$u_{2k-1}(t) := b_{2k-1}^2\left(\frac{t}{2}\right),$$
$$u_{2k}(t) := b_{2k}^2\left(\frac{t}{2}\right).$$

- *Continuous-time finite Lotka-Volterra equation,*

$$\frac{du_k}{dt} = u_k(u_{k+1} - u_{k-1}), \quad k = 1, 2, \ldots, 2n-1,$$

with $u_0(t) \equiv 0$ and $u_{2n}(t) \equiv 0$.

τ Functions

- Change variables

$$u_k = \frac{\tau_{k+2}\tau_{k-1}}{\tau_{k+1}\tau_k}$$

so that

$$\frac{d\ln u_k}{dt} = \frac{d}{dt}\ln\frac{\tau_{k+2}}{\tau_{k+1}} - \frac{d}{dt}\ln\frac{\tau_k}{\tau_{k-1}}.$$

- A compatibility condition,

$$\frac{\tau_{k+2}\tau_{k-1}}{\tau_{k+1}\tau_k} = \frac{d}{dt}\ln\frac{\tau_{k+1}}{\tau_k},$$

or equivalently,

$$\frac{d\tau_k}{dt}\tau_{k+1} - \tau_k\frac{d\tau_{k+1}}{dt} + \tau_{k-1}\tau_{k+2} = 0.$$

Determinantal Solution

- Starting with $\tau_{-1} \equiv 0$, $\tau_0 \equiv 1$, $\tau_1(t) = 1$ and $\tau_2(t) = \psi(t)$,

$$\begin{aligned} \tau_3 &= \frac{d\psi}{dt}, \\ \tau_4 &= \det \begin{bmatrix} \psi & \psi^{(1)} \\ \psi^{(1)} & \psi^{(2)} \end{bmatrix}, \end{aligned}$$

- In general (Tsujimoto'95),

$$\begin{aligned} \tau_{2k-1} &= \overline{H}_{k-1,1}, \\ \tau_{2k} &= \overline{H}_{k,0}, \end{aligned}$$

where

$$\overline{H}_{k,j}(t) := \det \begin{bmatrix} \psi^{(j)} & \psi^{(j+1)} & \dots & \psi^{(j+k-1)} \\ \psi^{(j+1)} & \psi^{(j+2)} & \dots & \psi^{(j+k)} \\ \vdots & \vdots & & \vdots \\ \psi^{(j+k-1)} & \psi^{(j+k)} & & \psi^{(j+2k-2)} \end{bmatrix}, \quad j = 0 \text{ or } 1,$$

is the determinant of a $k \times k$ Hankel matrix.

SVD Solution

- The general solution to the Lotka-Volterra equation (Tsujimoto, Nakamura &Iwasaki, '01)

$$
\begin{aligned}
u_{2k-1}(t) &= \frac{\overline{H}_{k,1}(t)\overline{H}_{k-1,0}(t)}{\overline{H}_{k,0}(t)\overline{H}_{k-1,1}(t)}, \\
u_{2k}(t) &= \frac{\overline{H}_{k+1,0}(t)\overline{H}_{k-1,1}(t)}{\overline{H}_{k,1}(t)\overline{H}_{k,0}(t)}, \quad k = 1, 2, \ldots, n,
\end{aligned}
$$

- Assuming all derivatives of ψ are obtainable from elementary calculus,
  - In principle, all Hankel determinants can be calculated algebraically.
  - All quantities about $u_k(t)$ are now in the analytic form.
  - The SVD flow and, hence, the iterates from the SVD algorithm are representable in closed form.

Discrete Lotka-Voterra Equation

(HIrota, Tsujimoto & Imai, '93)

- A key discretization of the Lotka-Volterra equation is a particular Euler-type scheme (*symplectic Euler*) of the form,

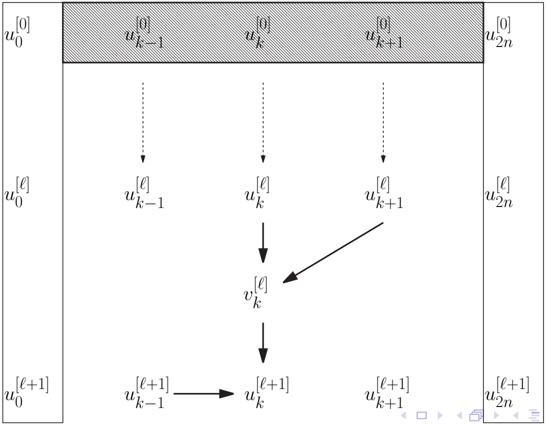
$$u_k^{[\ell+1]} = u_k^{[\ell]} + \delta \left(u_k^{[\ell]} u_{k+1}^{[\ell]} - u_k^{[\ell+1]} u_{k-1}^{[\ell+1]} \right).$$

  - $u_k^{[\ell]} \approx u_k(\ell\delta)$.
  - Boundary conditions $u_0^{[\ell]} \equiv 0$ and $u_{2n}^{[\ell]} \equiv 0$ for all ℓ.
- A mixture of both explicit and implicit Euler methods to maintain integrability.
  - Closely resemble the progressive qd algorithm (Rutishauser, '60) and the dqds algorithm (Fernando & Parlett, '94).

Variable Step Implementation

Iwasaki & Nakamura, '02, '04, '06)

$$u_k^{[\ell+1]}\left(1+\delta^{[\ell+1]}u_{k-1}^{[\ell+1]}\right)=\underbrace{u_k^{[\ell]}\left(1+\delta^{[\ell]}u_{k+1}^{[\ell]}\right)}_{v_k^{[\ell]}},$$



vdLV in Matrix Form

- For each ℓ, define

$$\begin{cases} q_i^{[\ell]} & := \quad \frac{1}{\delta^{[\ell]}}\left(1+\delta^{[\ell]}u_{2i-2}^{[\ell]}\right)\left(1+\delta^{[\ell]}u_{2i-1}^{[\ell]}\right), \quad i=1,\ldots,n, \\ e_j^{[\ell]} & := \quad \delta^{[\ell]}u_{2j-1}^{[\ell]}u_{2j}^{[\ell]}, \quad j=1,\ldots n-1. \end{cases}$$

- Assemble two $n\times n$ bidiagonal matrices,

$$L^{[\ell]} := \begin{bmatrix} q_1^{[\ell]} & 0 & & & 0 \\ 1 & q_2^{[\ell]} & & & \\ & & \ddots & & \\ & & \ddots & & \\ & & & 1 & q_n^{[\ell]} \end{bmatrix}, \quad R^{[\ell]} := \begin{bmatrix} 1 & e_1^{[\ell]} & & & \\ 0 & 1 & & & \\ & & \ddots & \ddots & \\ & & & & e_{n-1}^{[\ell]} \\ & & & & 1 \end{bmatrix}.$$

Progressive *qd* Algorithm
(Rutishauser, ’54, ’60)

- The vdLV is equivalent to the matrix equation

$$L^{[\ell+1]}R^{[\ell+1]} = R^{[\ell]}L^{[\ell]} - \underbrace{\left(\frac{1}{\delta^{[\ell]}} - \frac{1}{\delta^{[\ell+1]}}\right)}_{\text{built-in shift?}} I_n.$$

What Is up There?

- Introduce the tridiagonal matrix

$$Y^{[\ell]} := L^{[\ell]} R^{[\ell]} - \frac{1}{\delta^{[\ell]}} I_n.$$

- Can rewrite

$$Y^{[\ell]} = \begin{bmatrix} w_1^{[\ell]} & w_1^{[\ell]} w_2^{[\ell]} & 0 & & 0 \\ 1 & w_2^{[\ell]} + w_3^{[\ell]} & w_3^{[\ell]} w_4^{[\ell]} & & \\ & & \ddots & & \\ & & \ddots & & \\ & & \ddots & & \\ 0 & & & & w_{2n-3}^{[\ell]} w_{2n-2}^{[\ell]} \\ 0 & & & 1 & w_{2n-2}^{[\ell]} + w_{2n-1}^{[\ell]} \end{bmatrix},$$

  - $w_k^{[\ell]} := u_k^{[\ell]} \left(1 + \delta^{[\ell]} u_{k-1}^{[\ell]}\right).$

dqds Algorithm
(Fernando & Parlett, '94)

- The vdLV is equivalent to the similarity relationship

$$Y^{[\ell+1]} = R^{[\ell]} Y^{[\ell]} {R^{[\ell]}}^{-1}$$

  - $\{Y^{[\ell]}\}_\ell$ are isospectral.

New Integrals

- Can ensure $w_k^{[\ell]} > 0$ as long as $u_k^{[0]} > 0$ and $\delta^{[\ell]} > 0$, which can easily be achieved.
- Can symmetrize $Y^{[\ell]}$ via

$$Y_S^{[\ell]} := {D^{[\ell]}}^{-1} Y^{[\ell]} D^{[\ell]},$$

  - Easy to see that

$$D^{[\ell]} := \mathrm{diag}\left\{ \prod_{i=1}^{n-1} \sqrt{w_{2i-1}^{[\ell]} w_{2i}^{[\ell]}}, \prod_{i=2}^{n-1} \sqrt{w_{2i-1}^{[\ell]} w_{2i}^{[\ell]}}, \ldots, \sqrt{w_{2n-3}^{[\ell]} w_{2n-2}^{[\ell]}}, 1 \right\}.$$

Singular Values Deviations

- $Y_S^{[\ell]}$ enjoys a Cholesky decomposition

$$Y_S^{[\ell]} = {B^{[\ell]}}^\top B^{[\ell]},$$

with

$$B^{[\ell]} := \begin{bmatrix} \sqrt{w_1^{[\ell]}} & \sqrt{w_2^{[\ell]}} & & & & \\ 0 & \sqrt{w_3^{[\ell]}} & \sqrt{w_4^{[\ell]}} & & & \\ & & & \ddots & & \\ & & & \ddots & & \\ & & & & \sqrt{w_{2n-3}^{[\ell]}} & \sqrt{w_{2n-2}^{[\ell]}} \\ & & & & & \sqrt{w_{2n-1}^{[\ell]}} \end{bmatrix}.$$

- Singular values of the bidiagonal matrices $\{B^{[\ell]}\}_\ell$ derived from the vdLV are invariant.

Initial Correction

- Want singular values of a given matrix B_0.
- Constant drift.
- Choose initial values for the vdLV by

$$u_k^{[0]} := \frac{b_k(0)^2}{1 + \delta^{[0]} u_{k-1}^{[0]}}, \quad k = 1, 2, \ldots, 2n - 1.$$

Convergence and Stability
(Nakamura, ’06)

- With *any* step sizes $\delta^{[\ell]} > 0$,
  - $\{u_1^{[\ell]}, u_3^{[\ell]}, \ldots, u_{2n-1}^{[\ell]}\}_\ell$ converges to the squares of singular values of B_0 in descending order.
  - $u_{2k}^{[\ell]}$ converges to 0.
- Numerical stability.
  - No subtraction is involved.
  - All quantities are bounded by $\|B_0\|$.
- Greedy thoughts:
  - What is the convergence rate?
  - How to speed up convergence?

Constant Step Size $\delta^{[\ell]} \equiv \delta$

- Convergence is linear with asymptotic convergence factor

$$\alpha = \max_{k=1,\ldots,n-1} \frac{\sigma_{k+1} + \frac{1}{\delta}}{\sigma_k + \frac{1}{\delta}},$$

  - $\sigma_1 > \sigma_2 > \ldots > \sigma_n$ are the singular values of B_0.
  - Larger step sizes might reduce the value of α, but only to a certain extent.
- There is a built-in shift, but
  - Disappear with constant step size.
  - Become less effective with larger variable step sizes.
  - Need ”true" shift to make the *vdLV* algorithm efficient.

True Shifts

- The true shift really needed should be of the form

$$\overline{B}^{[\ell]^\top}\overline{B}^{[\ell]} = B^{[\ell]^\top}B^{[\ell]} - \theta^{[\ell]^2},$$

while keeping bidiagonal form

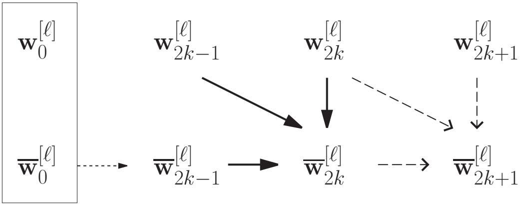
$$\overline{B}^{[\ell]} := \begin{bmatrix} \sqrt{\overline{w}_1^{[\ell]}} & \sqrt{\overline{w}_2^{[\ell]}} & & & & \\ 0 & \sqrt{\overline{w}_3^{[\ell]}} & \sqrt{\overline{w}_4^{[\ell]}} & & & \\ & & & \ddots & & \\ & & & \ddots & & \\ & & & & \sqrt{\overline{w}_{2n-3}^{[\ell]}} & \sqrt{\overline{w}_{2n-2}^{[\ell]}} \\ & & & & & \sqrt{\overline{w}_{2n-1}^{[\ell]}} \end{bmatrix}.$$

Computation Scheme

- Nonlinear relationship:

$$\begin{cases} \overline{w}_{2k}^{[\ell]} + \overline{w}_{2k+1}^{[\ell]} = w_{2k}^{[\ell]} + w_{2k+1}^{[\ell]} - {\theta^{[\ell]}}^2, \\ \overline{w}_{2k-1}^{[\ell]} \overline{w}_{2k}^{[\ell]} = w_{2k-1}^{[\ell]} w_{2k}^{[\ell]}, \end{cases} \quad k = 0, \ldots, n-1,$$

with $\overline{w}_0^{[\ell]} = w_0^{[\ell])} \equiv 0$.



mdLV Algorithm

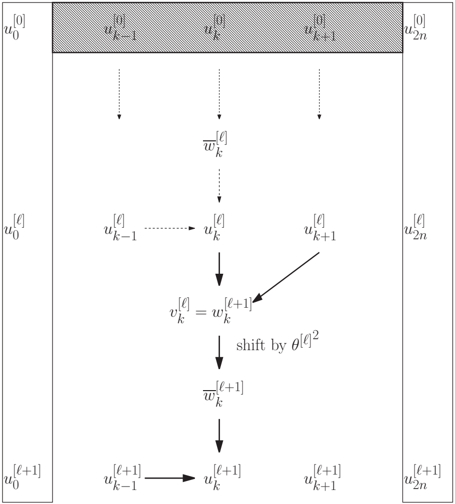
- Without shift,

$$u_k^{[\ell+1]} = \frac{w_k^{[\ell+1]}}{1 + \delta^{[\ell+1]} u_{k-1}^{[\ell+1]}},$$

- With shift,

$$u_k^{[\ell+1]} = \frac{\overline{w}_k^{[\ell+1]}}{1 + \delta^{[\ell+1]} u_{k-1}^{[\ell+1]}}.$$

mdLV Implementation



Conclusion

- Powerful discrete dynamical systems such as the *QR* algorithm and the *SVD* algorithm do have their continuous counterparts.
  - These differential systems often arise from seemingly rather distinct fields of disciplines.
- Diverse topics, such as soliton theory, integrable systems, continuous fractions, τ functions, orthogonal polynomials, the Sylvester identity, moments, and Hankel determinants, can all play together, intertwine, and eventually lead to the fact that the eigenvalues and the singular values of a given matrix can be expressed as the limit of some closed-form formulas!
- A careful discretization of a continuous dynamical system may indeed lead to an effective numerical algorithm.
  - By a “careful discretization", it is critical that the discrete scheme maintains its complete integrability.
  - A great many details such as shift strategies and implementation tactics also demand considerable attention.
- Classical SVD algorithm $\Rightarrow$ Lotka-Volterra equation $\Rightarrow$ *dLV* scheme $\Rightarrow$ *dqds* algorithm $\Rightarrow$ a brand new *mdLVs*.