Part1: Learning time-invariant dynamical systems.

Sadanand Vithal Giraddi
School of Computer Science
University of Lincoln
Lincoln, United Kingdom
25895786@students.lincoln.ac.uk

***Abstract*— In this assessment, the literature review has been conducted on Part1: for the learning time-invariant dynamical systems. And also stated the definition of a time-invariant dynamical system with its advantages and disadvantages compared with time-dependent dynamical systems. And also depicted the literature survey on learning time-invariant dynamical systems and the methods like Gaussian mixture regression for extracting the features, Stable Estimator of Dynamical systems (SEDS), algorithms like pi2 policy search algorithm and also noted how the stability and convergence have been obtained in different fields of application for the research papers noted down in the reference.**

Keywords*— *Time-invariant dynamical system, Gaussian mixture regression (GRE), Stable Estimator of Dynamical systems (SEDS).

I. Introduction

Time-invariant dynamical systems: A dynamical system is a system that is described by the Lagrangian function(L). It is the function taking the time, position, and velocity as inputs, so it is written as L(t,q,q˙), where t is called the time variable, q is called as the position of the variable, q˙ is called as the velocity of the variable.
It is said that the dynamical System is time-variant which means ∂L/∂t=0.
The desired properties of dynamical systems are Stability and Convergence, where the stability is the ability of systems to remain in a steady-state position and while convergence is the ability of systems to reach in a steady-state position [1]. And these are achieved with the help of the Lyapunov theorem:

Lyapunov function V(x), x 7→ R where,

V(x*)=0, for the stability in the steady-state position.

V(x) > 0∀x ∈ R,

∂V/∂t=0, for convergence [1].

Gaussian Mixture Regression: It is an approach for regression analysis based on the use of the model. It is the model which expresses the dataset by using multiple types of Gaussian distributions.
Gaussian mixture regression models are non-linear motion models used for learning complex demonstrations such as motion demonstrations [1][2]. Nonlinear features in DS representations are often created using Gaussian basis functions in one of two ways. The first method involves using a unidimensional input for the basis functions, such as time or an auxiliary phase variable [2].

Stable Estimator of Dynamical systems (SEDS):
The stable estimator of dynamical systems for learning the parameters of the dynamic systems for ensuring the motion that follows demonstration while reaching and stopping at the target [5]. And it is a powerful method for tackling the large challenge in the dynamical systems.

<u>Mathematically:</u>

Giving the train set from the demonstration
D {x1, x˙1, x2, x˙2, ..., xn, x˙n},
learn the mapping of f () such as ˙x = f (x)

<u>Using Guassian Mixture Regression:</u>
p(˙x|x) = PN n on(x)N (˙x|x; mn x˙|xΣ n x˙|x)
x˙ = f (x) = PN n=1 on(x)mn x˙|x

<u>Guarantee stability is implemented using Lyapunov Theory</u> [6][5]

V = 1/2 x T x 1 V(x∗) = 0, stability for steady-state

V(x) > 0∀x ∈ < 0, for the convergence

dV/dt <0, for the convergence

<u>Finding the parameters of the Gaussian Mixture Model which Minimize MSE:</u>

min θ J(θ) = 1 2T X N n=1 X Tn t=0 kx˙pred t,n − x˙ t,n demok 2

The stable estimator of dynamical systems constrains of :

bk = −Ak x ∗

Ak + (Ak) T ≺ 0

Σk >0

0 < ok ≤ 1

ΣK k=1 ok = 1

Where k= No of Gaussian Components
x ∗=Desired point of the stability
bk= Bias term for each Gaussian Regression line
Ak = Σk ˙ξξ(Σk ξ) −1 Slope term of each Gaussian Regression line
Ok=mixture of coefficient [6]

Advantages of the time-dependent dynamical system:

- External perturbations are inherently robust.
- It makes the robot to re-act instantly from the face of perturbations.
- Ensures trajectories for all the convergence.
- SEDS are time-independent dynamical systems.
- In SEDS, any number of demonstrations are used to model the robust motions.
- In SEDS, the generalization is not limited to any areas for demonstrations.
- The successive quadratic Programming approach have more advantages for the general-purpose solvers.

Disadvantages of the time-dependent dynamical system:

- Learning policies that can be applied over broad parts of the state space is difficult because trajectories of multiple initial states increase with the rollouts.
- Stability analysis for particular cases only exists for the nonlinear dynamical systems.
- In Dynamic Movement primitive, the canonical phase variable for DMP is explicitly time-dependent.
- In DMP, it learns the robust motions only from one demonstration.
- In DMP, the generalization is limited to areas that are close to the demonstrations.

II. Literature Survey

Learning time-invariant dynamical system through learning motions from the demonstrations and rewards based on policies, In reinforcement learning for learning the motions based on the policies is more challenging. In this paper, they used Gaussian mixture regression to extract the parameter as policy. The resulting parameter acts as a form of a non-linear time-invariant dynamical system. And used a time-invariant dynamical system as policy as the parameter for the PI2 policy search algorithm. They adapted this algorithm for time-variant motion representation. And also used two novel parameters i.e sample model is considered as the parameter for achieving the Uniform exploration in the state space position and the other parameter they considered is exploring the stability of the motion model. With this, the state-dependent stiffness profile is also learned to reference trajectory and both parameters are used in the variable impedance control architectures, later this architecture is validated for learning in the experiment of digging task by using the KUKA LWR as the platform. In this they shape the exploration noise for obtaining isotropic and homogeneous exploration in task space and also with this preserve the stability properties of the policy.

Algorithm 1 PI^2 for GMR parameterized policy

Given
- a set of demonstration trajectories *Demos*
- the covariance $\boldsymbol{\Sigma}_\epsilon$ of the exploration noise $\boldsymbol{\epsilon}$
- a cost function composed of:
  - a terminal cost term $\Phi(\boldsymbol{x}_{t_N})$
  - an immediate cost term $q(\boldsymbol{x}_{t_i})$
  - a quadratic control cost specified by the control matrix $\boldsymbol{R}$
- a transition function $\boldsymbol{f}(x)$ ($= \mathbf{0}$ or given by Eq. 12)

procedure INITIALIZE(*Demos*)
  GMR ← E- M(*Demos*)
  for k ← 1, N_G **do**
    $\boldsymbol{\theta}^k = [{A_{1,1}^k}^* \quad {A_{1,2}^k}^* \quad {A_{2,1}^k}^* \quad {A_{2,2}^k}^* \quad b_1^{k*} \quad b_2^{k*}]$
  end for
  $\boldsymbol{\theta} = [\boldsymbol{\theta}^1 \quad \boldsymbol{\theta}^2 \quad \dots \quad \boldsymbol{\theta}^{N_G}]$
end procedure
procedure POLICY SEARCH($\boldsymbol{\theta}$, Cost function)
  while Cost has not converged **do**
    $\alpha \leftarrow \max(\frac{1}{n_{iter}}, 0.1)$
    for $r \leftarrow 1, N_r$ **do**
      $\boldsymbol{\theta}^r \leftarrow \boldsymbol{\theta} + \boldsymbol{\epsilon}^r, \quad \boldsymbol{\epsilon}^r \in \mathcal{N}(0, \alpha \boldsymbol{\Sigma}_\epsilon)$
      $\boldsymbol{\tau}^r \leftarrow [\boldsymbol{x}_{t_1} \quad \dots \quad \boldsymbol{x}_{t_N}]_{\boldsymbol{\theta}^r}$ ▷ Sample trajectory
      $\boldsymbol{M}_{t_i,r} \leftarrow \boldsymbol{R}^{-1}\boldsymbol{G}_{t_i,r}^T(\boldsymbol{G}_{t_i,r}\boldsymbol{R}^{-1}\boldsymbol{G}_{xt_i}^T)^{-1}\boldsymbol{G}_{t_i,r}$
      $\Upsilon_{t_i,r} \leftarrow (\boldsymbol{\theta} + \boldsymbol{M}_{t_i,r}\boldsymbol{\epsilon}_{t_j,r})^T \boldsymbol{R}(\boldsymbol{\theta} + \boldsymbol{M}_{t_i,r}\boldsymbol{\epsilon}_r)$
      $S(\boldsymbol{\tau}^r) \leftarrow \Phi_{x_{t_N},r} + \sum_{j=0}^{N_t} q_{xt_j,r} + \frac{1}{2}\sum_{j=0}^{N_t} \Upsilon_{t_i,r}$
    end for
    $\lambda \leftarrow \frac{\max_r S(\boldsymbol{\tau}^r) - \min_r S(\boldsymbol{\tau}^r)}{10}$
    for $r \leftarrow 1, N_r$ **do**
      $P(\boldsymbol{\tau}^r) \leftarrow \frac{e^{-\frac{1}{\lambda}S(\boldsymbol{\tau}^r)}}{\sum_{r=1}^{N_r} e^{-\frac{1}{\lambda}S(\boldsymbol{\tau}^r)}}$
    end for
    $\delta\boldsymbol{\theta} \leftarrow \sum_{r=1}^{N_r} P(\boldsymbol{\tau}^r)\boldsymbol{\epsilon}^r$ ▷ Compute update
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \delta\boldsymbol{\theta}$ ▷ Update parameter vector
  end while
end procedure

Figure1: Policy search algorithm [2].

They considered two initial states for policy evaluation. This parameter is used to learn the behaviors of the state space. Two trajectories is evaluated with the cost function. To obtain the rollout the costs are added.
The variable impedance of the control architecture:

The damping is added with a simple point mass for tracking the trajectories.
xti = xti−1 + vti dt ,

vti = vti−1 + ati dt ,

ati = (uti − vti−1 · damping) /mass .

where xti is the position and vti is the velocity and ati is the acceleration of point mass at the time ti.

The KUKA LWR is having 7 degrees of freedom of the robotic arm. And the test contains shoveling gravel, A Barrett holds a shovel equipped with a 6-axis force/torque sensor securely in place. Hand mounted at the LWR's end. And the proposed time-invariant motion representation is suited for task manipulation with D<7 (D= Dimension). If more degrees of freedom is considered then the adapted reinforcement learning increases with a negative impact on the rate of convergence [2].

Neural networks are used for controlling and stabilizing the nonlinear dynamical system. In this paper, with the help of neural networks, they designed the viable controllers resulting in the nonlinear control theory. This paper considered for stabilization problem of the dynamical system around the equilibrium point when the state is accessible for the system. Neural networks used for adaptive control for nonlinear systems. And discrete-time dynamical system uses the below equations:
x(k+1)=f[x(k),u(k)] and y(k) = h[x(k)] where this equations represents input and output state of the system at time k.
In control, the input u(k) is determined for behaving in the system. And in the adaptive control, the f and h are assumed to be unknown, and to make the problem easy certain

assumptions should be considered like controllability and observability properties.
Liapunov's stability theorem: if V is the function of the equilibrium state x=0, then it is stable. The artificial neural network is used in which the neuron will refer to the operator and these neurons are organized in the form of layers l=0,1,…L and at l neuron layer it receives the inputs from neurons, and l-1 layer is called as feedforward neural network. In control, the class of systems is small for feedback linearizable. And stabilization is done using the linear controllers and in this, these are stabilized around the origin.
The methods proposed in this paper are practically viable and this is demonstrated by simulations based on the stabilization of the non-linear systems. The control is more complex when the system state is not accessible. And using the input and output data the control has been achieved [3].

Linear Time-Invariant Dynamical Systems Invertibility, the dynamical systems are inverted for determining the properties and existence of inverse and in terms of matrix construction. In this work, an inherent concept is introduced to the dynamical systems. Unless ideal differentiators are supplied, no inverse dynamical system can remove the number of integrations. The inverse is considered in terms of inherent integrations and the existing tests introduced are more complex. The existence tests proposed are at most half as complicated as Brockett and Mesarovic's, and the architecture is significantly more conceptually simple than Youla and Dorato's. And the results made it possible by recognizing the equivalence in the real sequential circuits. They appear to be useful to sensitivity, estimation, and game theory issues. The dynamical systems are defined on the real field because the stability of the system is valid over any of the fields or finite. The concept of inherent integration of a continuous dynamical system determines the parameter and tells the integrations that are associated with the input from the system. And from that, the differentiations are chosen to be made on the inverse of a system [4]. And when the inverse got existed the new construction appeared and it is simpler when it is compared to the visualization with Youla and Dorato and evaluated it [4].

Gaussian mixture models were implemented for learning stable nonlinear dynamical systems. In this work, they proposed the method to learn the discrete robot motions by using gaussian mixture models from the set of demonstrations. They model the motion for a time-invariant dynamical system and sufficient conditions for ensuring the global asymptotic stability for the target. They proposed the stable estimator of dynamical systems for learning the parameters of the dynamic systems for ensuring the motion that follows demonstration while reaching and stopping at the target. At the target, time-invariant and global asymptotic stability ensure that the system can react quickly and appropriately to perturbations encountered during motion and evaluated through the robot experiments of human handwriting motions. Point-to-point motions were considered for modeling as it provides a basic component for the robots to control. At first, they consider for pick and place task for reaching the item and moving to its target location and coming back to its original position. Using a library of 20 handwriting movements, they compared the performance of the suggested method to that of commonly used regression algorithms. From the evaluation table, SEDS likelihood achieved the stability and globally ensured. GMR did not achieve stability but it ensured for local.
They also tested the approaches in a variety of point-to-point robot tasks using two distinct robots. In all of the studies, the suggested approach was able to complete the experiments with good accuracy during replication.

Figure 2: Demonstrating the motions by using the motion sensor and teleoperating with the help of a hand [5].

For both SEDS-MSE and SEDS-likelihood were formulated using the nonlinear programming problem which are solved using optimization techniques. And they used the successive quadratic programming approach method for solving the constrained optimization problem.

Algorithm 1 Procedure to determine an initial guess for the optimization parameters

Input: $\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{t=0,n=1}^{T^n,N}$ and K

1: Run EM over demonstrations to find an estimate of π^k, μ^k, and Σ^k, $k \in 1..K$.
2: Define $\tilde{\pi}^k = \pi^k$ and $\tilde{\mu}_\xi^k = \mu_\xi^k$
3: Transform covariance matrices such that they satisfy the optimization constraints given by Eq. 20(b) and (c):

$$\begin{cases} \tilde{\Sigma}_\xi^k = \boldsymbol{I} \circ abs(\Sigma_\xi^k) \\ \tilde{\Sigma}_{\dot{\xi}\xi}^k = -\boldsymbol{I} \circ abs(\Sigma_{\dot{\xi}\xi}^k) \\ \tilde{\Sigma}_{\dot{\xi}}^k = \boldsymbol{I} \circ abs(\Sigma_{\dot{\xi}}^k) \\ \tilde{\Sigma}_{\xi\dot{\xi}}^k = -\boldsymbol{I} \circ abs(\Sigma_{\xi\dot{\xi}}^k) \end{cases} \qquad \forall k \in 1..K$$

where $\circ$ and $abs(.)$ corresponds to entrywise product and absolute value function, and $\boldsymbol{I}$ is a $d \times d$ identity matrix.
4: Compute $\tilde{\mu}_{\dot{\xi}}^k$ by solving the optimization constraint given by Eq. 20(a):

$$\tilde{\mu}_{\dot{\xi}}^k = \tilde{\Sigma}_{\dot{\xi}\xi}^k (\tilde{\Sigma}_\xi^k)^{-1} (\tilde{\mu}_\xi^k - \xi^*)$$

Output: $\boldsymbol{\theta}^0 = \{\tilde{\pi}^1..\tilde{\pi}^K; \tilde{\mu}^1..\tilde{\mu}^K; \tilde{\Sigma}^1..\tilde{\Sigma}^K\}$

Figure 3: Algorithm for determining the initial guess for the optimization parameters.

The performance improves with the analytic expression of the derivatives. As the above algorithm is used as a simple and efficient way for computing the guess for the initial guess and from the initial value solver tries optimizing the value to zero for minimizing the cost function J. and this algorithm yields a good local minimum value. And the validated the SEDS for estimating the dynamics of motions for the two platforms of the robot. They are the 7 degrees of freedom of humanoid robot (right hand) iCub. And six degrees of freedom Katana-T arm industrial robot. And compared the method with four alternative methods [5].

REFERENCES

[1]. Advanced Robotics, week9-lecture, professor-[Dr. Athanasios Polydoros].

[2]. Rey, J., Kronander, K., Farshidian, F., Buchli, J. and Billard, A., 2018. Learning motions from demonstrations and rewards with time-invariant dynamical systems based policies. *Autonomous Robots*, *42*(1), pp.45-64.

[3]. Levin, A.U. and Narendra, K.S., 1993. Control of nonlinear dynamical systems using neural networks: Controllability and stabilization. *IEEE Transactions on neural networks*, *4*(2), pp.192-206.

[4]. Sain, M. and Massey, J., 1969. Invertibility of linear time-invariant dynamical systems. *IEEE Transactions on automatic control*, *14*(2), pp.141-149.

[5]. Khansari-Zadeh, S.M. and Billard, A., 2011. Learning stable nonlinear dynamical systems with gaussian mixture models. *IEEE Transactions on Robotics*, *27*(5), pp.943-957.

[6]. Advanced Robotics, week10-lecture, professor-[Dr. Athanasios Polydoros].