Physics-based Learning for Aircraft Dynamics Simulation

Yang Yu[1], Houpu Yao[2], and Yongming Liu[3]

[1]*School for Engineering of Matter, Transport & Energy, Arizona State University, Tempe, AZ, 85281, USA*
yang.yu@asu.edu

[2]*School for Engineering of Matter, Transport & Energy, Arizona State University, Tempe, AZ, 85281, USA*
houpu.yao@asu.edu

[3]*School for Engineering of Matter, Transport & Energy, Arizona State University, Tempe, AZ, 85281, USA*
yongming.liu@asu.edu

ABSTRACT

The accurate prediction of flight trajectories is crucial for the real-time prognostics of air transportation system. However, the computation costs of predictions can be expensive or even prohibitive especially for a large number of aircrafts in the air traffic system. This study proposes the concept of physics-based learning, a hybrid approach based on data-driven learning and physical models, as a computationally efficient method for the simulation of aircraft dynamics. The physics-based learning integrates the underlying physics of dynamical systems into learning models such as neural networks to reduce the training and simulation costs. The application of physics-based learning for simulating aircraft dynamics is demonstrated using a recently introduced physics-aware network known as the deep residual recurrent neural network (DR-RNN) on a Boeing 747-100 aircraft. The aircraft dynamics are described using a six degrees-of-freedom aircraft model. The DR-RNN is first trained using the simulated responses of the aircraft and then the trained network is used to predict the response of aircraft under arbitrary control inputs and disturbances. The results show that the DR-RNN can accurately predict aircraft responses and has excellent extrapolation capabilities. Moreover, the DR-RNN exhibits superior computation efficiency compared with a classical numerical method, the fourth-order Runge-kutta method, highlighting its suitability in serving as surrogating models for aircraft dynamical systems.

1. INTRODUCTION

The prediction of aircraft trajectories through the simulation of aircraft dynamical systems provides important information for the risk assessment of air transportation safety. The aircraft dynamical system is governed by the equations of motion of the aircraft that can be solved using numerical methods. However, the computation costs of these operations can be expensive or even prohibitive especially for a large number of aircrafts in the air traffic system. Therefore, a computationally efficient method is needed for the simulation of aircraft dynamical systems.

Recurrent neural network (RNN) is a class of artificial neural networks where units are connected to form a directed graph. RNNs have achieved great success in many different areas such as language modeling, speech recognition, and time series prediction (Hinton et al., 2012; Graves, 2013). The general RNN architecture is illustrated in Figure 1. RNN uses its internal states (s_{t-1} shown in Figure 1) as memories to learn the time dependencies and thus it is capable to model the evolution of dynamical systems. One of the advantages of RNNs is that the weights of the network are shared across all time steps. Therefore, the same operations are performed at each step with different inputs, which significantly reduces the number of parameters during training. In the past, there have been applications on using RNNs as surrogate models to simulate dynamical systems (Trischler & D'Eleuterio, 2016). Nevertheless, a major challenge of using RNNs to simulate dynamical systems is the high training costs. This is because the learning using most RNNs is purely data-driven, which usually requires significant amount of training data and a large number of training parameters.

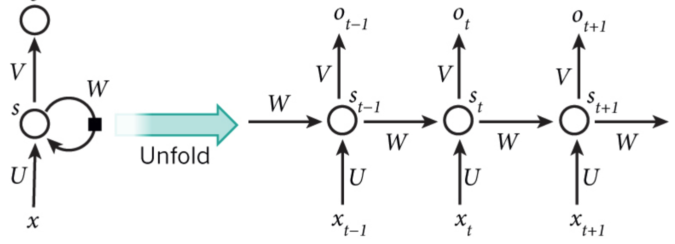


Figure 1. Illustration of the RNN architecture (Britz, 2015)

In this study, the concept of physics-based learning is proposed. Physics-based learning is a hybrid approach that utilizes both the data-driven learning and the underlying physics of dynamical systems to achieve more efficient learning and prediction. Specifically, the underlying physics of the dynamical system is integrated into the learning models such as RNNs to provide additional constraints for the learning and prediction of dynamical system behaviors. By doing so, the physics-based learning method is able to greatly reduce the training costs associated with purely data-driven methods. The trained model can serve as surrogate models for aircraft dynamical systems and thus reduce the high computation costs of directly solving the system. Furthermore, the integration of the physics will enhance the extrapolation performances of the learning model. This is considered as a desirable feature since the long-term responses of dynamical systems under arbitrary inputs are often of interest. Recently, a physics-aware RNN architecture known as the deep residual RNN (DR-RNN) was introduced (Kani & Elsheikh, 2017). The DR-RNN formulates an iterative scheme to minimize the residual function that is computed using the underlying physics of the dynamical system. In this study, the DR-RNN will be adopted to handle learning of aircraft dynamics.

The objective of this study is to introduce the physics-based learning for the simulation of aircraft dynamics. The aircraft dynamical system is represented as a six degrees of freedom (DOFs) model. The derivation of the equations of motion of the aircraft is presented. Then, the standard RNN architecture is briefly review, which leads to the introduction of the DR-RNN as a form of physics-based learning. To demonstrate the application, the DR-RNN is trained to learn the dynamical behavior of a large transport aircraft, the Boeing 747. The trained network is used to predict the responses of the aircraft under arbitrary control and disturbances, and the computation efficiency of the DR-RNN for simulating aircraft dynamics is analyzed.

2. PHYSICS-BASED LEARNING

2.1. Standard Recurrent Neural Networks

The standard RNN architecture can be written as (Goodfellow et al., 2016):

$$\boldsymbol{h}_t = \tanh\left(\boldsymbol{W}\boldsymbol{h}_{t-1} + \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{b}\right) \tag{1}$$

$$\boldsymbol{o}_t = \boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{c} \tag{2}$$

where $\boldsymbol{x}_t$ is the input at time instant *t*; $\boldsymbol{h}_t$ is the internal state at time instant *t*; $\boldsymbol{o}_t$ is the output at time instant *t* ; ***b*** and ***c*** are the biases of the RNN; ***W***, ***U***, and ***V*** are the weight parameters of the RNN. It can be seen that the current internal state is computed using the current input and the internal state at the previous time step which serves as the memory of the RNN. The parameters of RNN, i.e., $\boldsymbol{\theta} = \left[\boldsymbol{W}, \boldsymbol{U}, \boldsymbol{V}, \boldsymbol{b}, \boldsymbol{c}\right]$, are estimated by minimizing the loss function. For time series prediction, the loss function can be written as:

$$L(\boldsymbol{\theta}) = \frac{1}{TSN}\sum_{s=1}^{S}\sum_{n}^{N}\sum_{t=1}^{T}\left|\boldsymbol{y}_t^{pred} - \boldsymbol{y}_t^{true}\right| \tag{3}$$

where $\boldsymbol{y}_t^{pred}$ and $\boldsymbol{y}_t^{true}$ are the predicted and true responses of the dynamical system at time instant *t*, respectively; *T* is the number of time steps; *N* is the number of features (number of states of the dynamical system); *S* is the number of training samples. During training, the values of RNN parameters are updated using back-propagation through time (Werbos, 1990). However, it was found that the standard RNN architecture often has difficulties in learning long-term dependencies due to the vanishing or exploding gradient problem (Pascanu et al., 2013). To address this problem, gated RNN architectures such as long short-term memory (Hochreiter and Schmidhuber, 1997) and gated recurrent unit (Cho et al., 2014) were introduced.

2.2. Deep Residual Recurrent Neural Networks

For dynamical systems, the physics is reflected in their governing equations whose general form can be written as:

$$\frac{d\mathbf{y}}{dt} = f(t, \mathbf{y}) \tag{4}$$

where **y** is the state of the dynamical system. Traditionally, Eq. (4) can be solved either analytically or numerically to obtain the response of the dynamical system. For example, the state at time instant *t*+1 can be obtained using the implicit Euler method as:

$$\mathbf{y}_{t+1} = \mathbf{y}_t + h \times f(t+1, \mathbf{y}_{t+1}) \tag{5}$$

where *h* is the time step size. From Eq. (5), a residual function can be formulated as:

$$\mathbf{r}_{t+1} = \mathbf{y}_{t+1} - \mathbf{y}_t - h \times f(t+1, \mathbf{y}_{t+1}) \tag{6}$$

The DR-RNN architecture is designed to iteratively minimize the residual function given in Eq. (6) by stacking K network layers (Kani & Elsheikh, 2017):

$$\begin{aligned} \boldsymbol{y}_{t+1}^{(k)} &= \boldsymbol{y}_{t+1}^{(k-1)} - \boldsymbol{W} \circ \tanh(\boldsymbol{U}\boldsymbol{r}_{t+1}^{(k)}), \text{ for } k = 1 \\ \boldsymbol{y}_{t+1}^{(k)} &= \boldsymbol{y}_{t+1}^{(k-1)} - \frac{\eta_k}{\sqrt{G_k + \varepsilon}}\boldsymbol{r}_{t+1}^{(k)}, \text{ for } k > 1 \end{aligned} \tag{7}$$

where k is the layer number; $\boldsymbol{r}_{t+1}^{(k)}$ is the residual at time instant *t* +1 in the *k*th layer; the operator $\circ$ denotes element-wise multiplication; ***W***, ***U***, and η are the weight parameters of the DR-RNN; ε is a small number to avoid division by zero; and G_k is the exponentially decaying squared norm of the residual calculated as:

$$G_k = \gamma\left\|\boldsymbol{r}_{t+1}\right\|^2 + \zeta G_{k-1} \tag{8}$$

where γ and ζ are the fraction factors and their values are often set as 0.1 and 0.9, respectively (Tieleman & Hinton, 2012). In this study, the DR-RNN is implemented in Tensorflow^{TM} and the Adam optimizer (Kingma and Ba, 2014) is adopted to minimize the loss function given in Eq. (3). In this case, $\boldsymbol{y}_{pred}$ and $\boldsymbol{y}_{true}$ are three-dimensional tensors with the shape of [N, M, H] where N is the batch size; M is the number of time steps; and H is the number of states of the dynamical system. In addition, it can be seen from Eq. (7) that the DR-RNN is explicit in time with a constant computational cost at each time step.

3. AIRCRAFT DYNAMICS

3.1. Six DOFs Aircraft Model

In aircraft dynamics, the aircraft is usually assumed as a rigid body and a six DOFs aircraft model shown in Figure 2 can be adopted to simulate the aircraft behavior. Choosing the stability axes as the body axes, the equations of motion of the aircraft can be written as:

$$\begin{aligned} X &= m(\dot{u} + qw - rv) \\ Y &= m(\dot{v} + ru - pw) \\ Z &= m(\dot{w} + pv - qu) \end{aligned} \tag{9}$$

$$\begin{aligned} L &= I_{xx}\dot{p} + I_{xz}\dot{r} + qr(I_{zz} - I_{yy}) + pqI_{xz} \\ M &= I_{yy}\dot{q} + pr(I_{xx} - I_{zz}) + \left(r^2 - p^2\right)I_{xz} \\ N &= I_{zz}\dot{r} + I_{xz}\dot{p} + pr(I_{yy} - I_{xx}) - qrI_{xz} \end{aligned} \tag{10}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{11}$$

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{z}_e \end{bmatrix} = \left[T\right] \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{12}$$

where (*X*, *Y*, *Z*) are the axial, side, and normal forces applied to the aircraft, respectively; (*L*,*M*,*N*) are the rolling, pitching, and yawing moments applied to the aircraft, respectively; (*u*,*v*,*w*) are the linear velocities in the *x*, *y*, *z* directions, respectively; (*p*,*q*,*r*) are the roll, pitch and yaw rates of the aircraft, respectively; (ϕ, θ, φ) are the roll, pitch and yaw angles of the aircraft, respectively; (x_e, y_e, z_e) are the north, east positions and the negative altitude of the aircraft with respect to the earth, respectively; I_{xx}, I_{yy}, and I_{zz} are the moments of inertial about (*x*, *y*, *z*) axes, respectively; I_{xz} is the product of inertial; and [*T*] is the transformation matrix given as:

$$\begin{bmatrix} \cos\theta\cos\varphi & \sin\phi\sin\theta\cos\varphi - \cos\phi\sin\varphi & \cos\phi\sin\theta\cos\varphi + \sin\phi\sin\varphi \\ \cos\theta\sin\varphi & \sin\phi\sin\theta\sin\varphi + \cos\phi\cos\varphi & \cos\phi\sin\theta\sin\varphi - \sin\phi\cos\varphi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

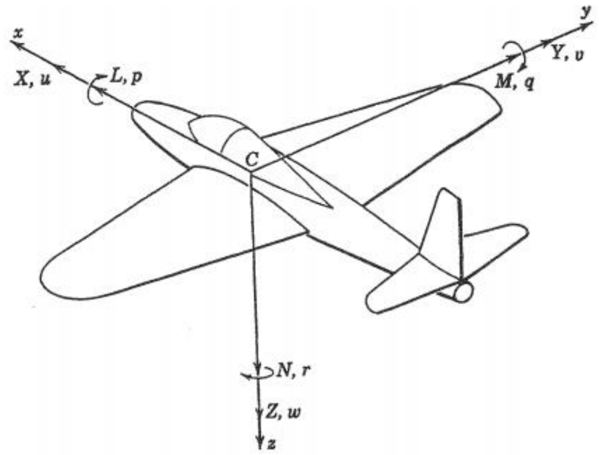


Figure 2. Six DOFs aircraft model (Pete, 2010)

3.2. Small-disturbance Theory

Eqs. (9) and (10) can be linearized under small perturbations to the an equilibrium flight condition (Etkin and Reid 1996). In this study, the reference (equilibrium) flight condition is assumed to be in longitudinal trim with no angular velocity, i.e., $v_0 = p_0 = q_0 = r_0 = \phi_0 = 0$. The stability axes are selected so that lift and drag are aligned with the *Z* and *X* axes, i.e., $w_0 = 0$. At reference flight condition, the aircraft is assumed to have velocity u_0 and pitch angle θ_0. Applying the above conditions and ignoring high-order terms in Eqs. (9) and (10), the longitudinal and lateral motions of the aircraft can be decoupled and linearized equations of motion can be represented in state space forms as:

1) For longitudinal motion:

$$\begin{bmatrix}\Delta\dot{u}\\ \dot{w}\\ \dot{q}\\ \Delta\dot{\theta}\end{bmatrix}=\begin{bmatrix}\frac{X_u}{m} & \frac{X_w}{m} & 0 & -g\cos(\theta_0)\\ \frac{Z_u}{m-Z_{\dot{w}}} & \frac{Z_w}{m-Z_{\dot{w}}} & \frac{Z_q+mu_0}{m-Z_{\dot{w}}} & -\frac{m\sin(\theta_0)}{m-Z_{\dot{w}}}\\ \frac{1}{I_{yy}}\left[M_u+\frac{M_{\dot{w}}Z_u}{m-Z_{\dot{w}}}\right] & \frac{1}{I_{yy}}\left[M_w+\frac{M_{\dot{w}}Z_w}{m-Z_{\dot{w}}}\right] & \frac{1}{I_{yy}}\left[M_q+\frac{M_{\dot{w}}(Z_q+mu_0)}{m-Z_{\dot{w}}}\right] & \frac{M_{\dot{w}}mg\sin(\theta_0)}{I_{yy}(m-Z_{\dot{w}})}\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}\Delta u\\ w\\ q\\ \Delta\theta\end{bmatrix} \tag{13}$$

$$+\begin{bmatrix}\frac{X_{\delta_e}}{m} & \frac{X_{\delta_{th}}}{m}\\ \frac{Z_{\delta_e}}{m-Z_{\dot{w}}} & \frac{Z_{\delta_{th}}}{m-Z_{\dot{w}}}\\ \frac{M_{\delta_e}}{I_{yy}}+\frac{M_{\dot{w}}Z_{\delta_e}}{I_{yy}(m-Z_{\dot{w}})} & \frac{M_{\delta_{th}}}{I_{yy}}+\frac{M_{\dot{w}}Z_{\delta_{th}}}{I_{yy}(m-Z_{\dot{w}})}\\ 0 & 0\end{bmatrix}\begin{bmatrix}\Delta\delta_e\\ \Delta\delta_{th}\end{bmatrix}$$

2) For lateral motion:

$$\begin{bmatrix}\dot{v}\\ \dot{p}\\ \dot{r}\\ \dot{\phi}\\ \dot{\varphi}\end{bmatrix}=\begin{bmatrix}\frac{Y_v}{m} & \frac{Y_p}{m} & (\frac{Y_r}{m}-u_0) & g\cos(\theta_0) & 0\\ \frac{L_v}{I'_x}+I'_{xz}N_v & \frac{L_p}{I'_x}+I'_{xz}N_p & \frac{L_r}{I'_x}+I'_{xz}N_r & 0 & 0\\ I'_{xz}L_v+\frac{N_v}{I'_z} & I'_{xz}L_p+\frac{N_p}{I'_z} & I'_{xz}L_r+\frac{N_r}{I'_z} & 0 & 0\\ 0 & 1 & \tan(\theta_0) & 0 & 0\\ 0 & 0 & \sec(\theta_0) & 0 & 0\end{bmatrix}\begin{bmatrix}v\\ p\\ r\\ \phi\\ \varphi\end{bmatrix}+\begin{bmatrix}\frac{Y_{\delta_a}}{m} & \frac{Y_{\delta_r}}{m}\\ \frac{L_{\delta_a}}{I'_x}+I'_{xz}N_{\delta_a} & \frac{L_{\delta_r}}{I'_x}+I'_{xz}N_{\delta_r}\\ I'_{xz}L_{\delta_a}+\frac{N_{\delta_a}}{I'_z} & I'_{xz}L_{\delta_r}+\frac{N_{\delta_r}}{I'_z}\\ 0 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}\delta_a\\ \delta_r\end{bmatrix} \tag{14}$$

$$I'_x=\left(I_{xx}I_{zz}-I_{xz}{}^2\right)/I_{zz}$$

and $I'_z=\left(I_{xx}I_{zz}-I_{xz}{}^2\right)/I_{xx}$

$$I'_{xz}=I_{xz}/\left(I_{xx}I_{zz}-I_{xz}{}^2\right)$$

where $\Delta\delta_e$, $\Delta\delta_{th}$, δ_a, and δ_r are the elevator deflection, thrust, aileron deflection, and rudder deflection of the aircraft, respectively. In Eqs. (13) and (14), the notation of force or moment with a subscript of the aircraft state or control input denotes the aerodynamic stability or control derivative. For example, X_u denotes the derivative of the aerodynamic force in the *x* direction with respect to the linear velocity *u* of the aircraft. The complete state and control vectors of the aircraft are expressed as:

$$\boldsymbol{x}=\begin{bmatrix}u & w & q & \theta & v & p & r & \phi & \varphi\end{bmatrix}^T \tag{15}$$

$$\boldsymbol{\delta}=\begin{bmatrix}\Delta\delta_e & \Delta\delta_{th} & \delta_a & \delta_r\end{bmatrix}^T \tag{16}$$

4. Demonstration

4.1. Aircraft Data

The application of physics-based learning for simulating aircraft dynamics is demonstrated here using a large transport aircraft, the Boeing 747-100 (Figure 3). In this study, the reference flight condition is set as steady level flight at 12,192 m (40,000 ft) and Mach 0.8. The aircraft properties and aerodynamic derivatives of the Boeing 747-100 at the reference flight condition are given in Table 1, Table 2 and Table 3.

Figure 3. Boeing 747-100 (Wikipedia, 2018)

Table 1. Boeing 747-100 data at reference flight condition (Etkin and Reid, 1996).

$W = 2.83176 \times 10^6\ N$	$S = 511.0\ m^2$
$\bar{c} = 8.234\ m$	$b = 59.64\ m$
$I_{xx} = 0.247 \times 10^8\ kg\ m^2$	$I_{yy} = 0.449 \times 10^8\ kg\ m^2$
$I_{zz} = 0.673 \times 10^8\ kg\ m^2$	$I_{xz} = -0.212 \times 10^7\ kg\ m^2$
$u_0 = 235.9\ m/s$	$\rho = 0.3045\ kg/m^3$
$\theta_0 = 0$	

Note: W is the weight of the aircraft; S is the wing area; b is the wing span; $\bar{c}$ is the standard mean chord of the wing; u_0 is the reference flight velocity; ρ is the air density.

Table 2. Longitudinal stability and control derivatives of Boeing 747-100 at reference flight condition (Etkin and Reid, 1996).

$\partial()/\partial()$	$X(N)$	$Z(N)$	$M(N\cdot m)$
$u(m/s)$	-1.98×10^3	-2.595×10^4	-1.593×10^4
$w(m/s)$	4.025×10^3	-9.030×10^4	-1.563×10^4
$q(rad/s)$	0	-4.524×10^5	-1.521×10^7
$\dot{w}(m/s^2)$	0	-1.909×10^3	-1.702×10^4
$\delta_e(rad)$	-16.5299	-1.5794×10^6	-5.204×10^7
δ_{th}	849528	0	0

Note: δ_{th} is a dimensionless number between 0 and 1.

Table 3. Lateral stability and control derivatives of Boeing 747-100 at reference flight condition (Etkin and Reid, 1996).

$\partial()/\partial()$	$Y(N)$	$L(N\cdot m)$	$N(N\cdot m)$
$v(m/s)$	-1.610×10^4	-3.062×10^5	-2.131×10^5
$p(rad/s)$	0	-1.076×10^7	-1.330×10^6
$r(rad/s)$	0	-9.925×10^6	-8.934×10^6
$\delta_a(rad)$	0	-3.5323×10^6	-5.0945×10^4
$\delta_r(rad)$	-4.9616×10^5	-1.8013×10^6	-3.2457×10^7

4.2. DR-RNN Training

With the aircraft data given in the previous section, the equations of motion can be solved to obtain the aircraft responses. In this study, the training data is obtained by solving Eqs. (13) and (14) given random initial disturbances and specified control inputs. For the longitudinal motion, the training control input is given as a one degree elevator step function and for the lateral motion, the training control input is given as one degree rudder step function. For both longitudinal and lateral motions, random initial disturbances following a uniform distribution from -0.05 to 0.05 are used when generating the training data. Five hundred samples of training data for the duration of 10 seconds were generated and used to train the DR-RNN. It should be noted that since the longitudinal and lateral motions of the aircraft are decoupled, two DR-RNNs are used to learn the longitudinal and lateral dynamics of the aircraft, respectively. The training step sizes for longitudinal and lateral motions are 0.1 s and 0.05 s, respectively. The training was implemented in Tensorflow which calculates the gradients symbolically based on the computation graph. Also, it is noted that for the training of neural networks, large differences of feature scales could cause training difficulties (Ba et al., 2016). Therefore, the linear velocities of the aircraft are normalized with respect to the reference flight velocity before the training and the state vector of the aircraft becomes:

$$\mathbf{x} = \begin{bmatrix} \frac{u}{u_0} & \alpha & q & \theta & \beta & p & r & \phi & \varphi \end{bmatrix}^T \quad (17)$$

where u/u_0 is the normalized velocity in the x direction; $\alpha \cong \tan\alpha = w/u_0$ is the angle of attack; and $\beta \cong \tan\beta = v/u_0$ is the sideslip angle. During training, the training data is divided into batches with a batch size of 16 and it was found that using a two-layer DR-RNN can reach sufficient accuracy. The loss function for longitudinal and lateral motions converged to around 5.5e-8 and 2.2e-5 after 20 minutes of training on GPU device Nvidia Gefore GTX 1080. Figure 4 shows the convergence of the loss function for the lateral motion of aircraft.

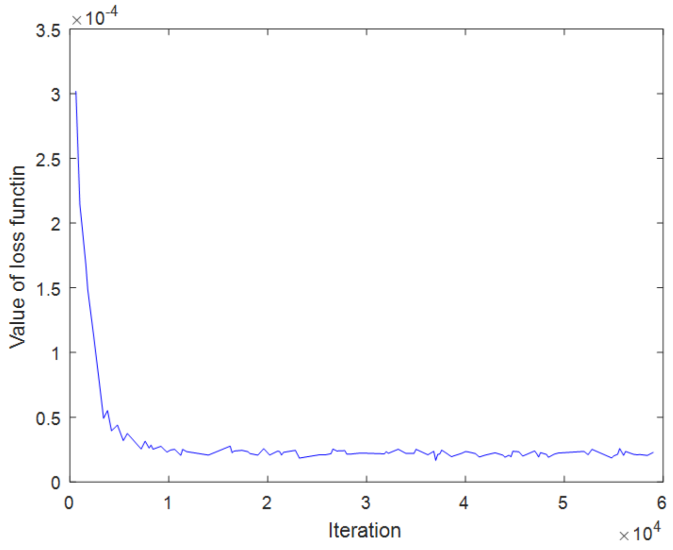


Figure 4. Convergence of loss function for the lateral motion (one batch for each iteration)

4.3. Prediction using Trained DR-RNN

The trained DR-RNNs are adopted to predict the responses of aircraft under arbitrary disturbances and control inputs. For demonstration purposes, a total of five prediction cases are considered in this study and the description of these cases is given in Table 4: (1) Cases 1 and 2 involve only control input and disturbance to the longitudinal motion of the aircraft; (2) Cases 3 and 4 involve only control input and disturbance to the lateral motion of the aircraft; (3) Case 5 involves control inputs to both longitudinal and lateral motions. The prediction duration is chosen to reflect the oscillations of aircraft responses before the system reaches the steady state.

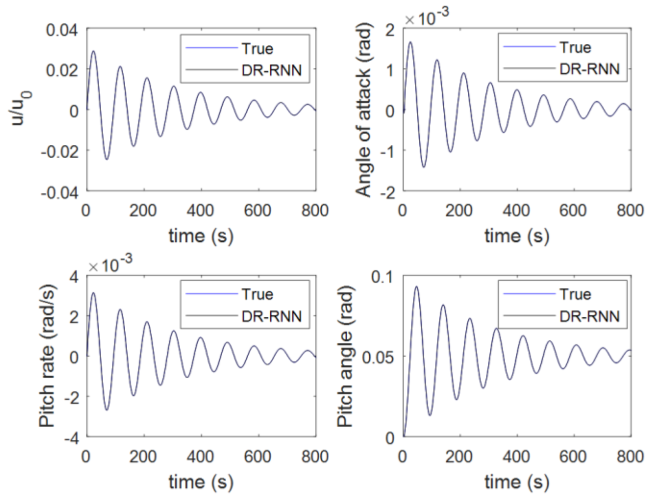
Table 4. Description of cases considered for prediction.

Case number	Initial disturbance to states	Control inputs				Prediction duration
		Elevator	Thrust	Aileron	Rudder	
Case 1	q_d =0.15 rad/s	N.A.	N.A.	N.A.	N.A.	800 s
Case 2	N.A.	N.A.	1/6 step input	N.A.	N.A.	800 s
Case 3	p_d =0.12 rad/s	N.A.	N.A.	N.A.	N.A.	200 s
Case 4	N.A.	N.A.	N.A.	1 degree 2-sec impulse input	N.A.	200 s
Case 5	N.A.	1 degree 5-sec doublet input	1/4 50-sec input	1 degree 2-sec doublet input	1 degree 2-sec impulse input	800 s

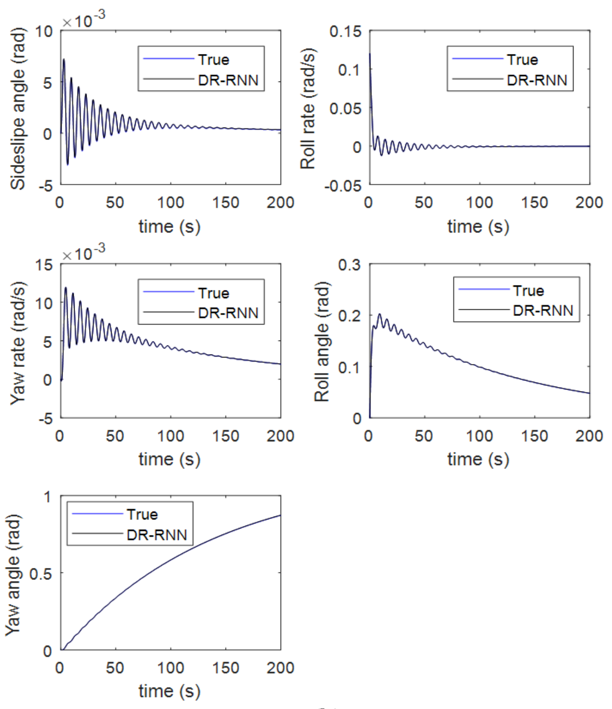
Figure 5 shows the predicted responses of the aircraft for Cases 2 and 3. It can be observed that the predicted responses matched very well with the true responses. Table 5 gives the prediction errors calculated using Eq. (3) for all prediction cases. The small error values suggest that the DR-RNN is able to accurately predict the response of aircraft under different control inputs and disturbances. Furthermore, the prediction results indicate that the DR-RNN has extrapolation capabilities since: (1) the prediction duration is much larger than the training data duration of 10 s; (2) the control inputs given for the prediction are different from those given during training. The good extrapolation performances of the DR-RNN are attributed to the integration of the underlying physics of the dynamical system into the learning model.

Table 5. Prediction errors of considered cases.

Case number	Prediction error
Case 1	7.75e-7
Case 2	3.17e-7
Case 3	1.42e-4
Case 4	7.19e-5
Case 5	2.63e-4



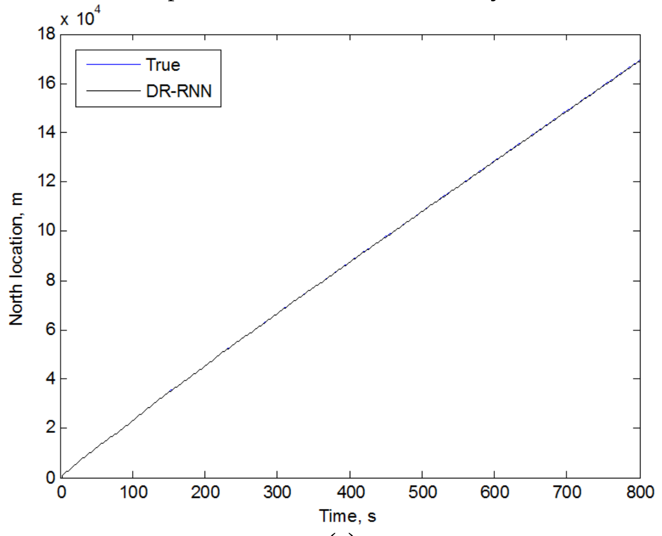
(a)



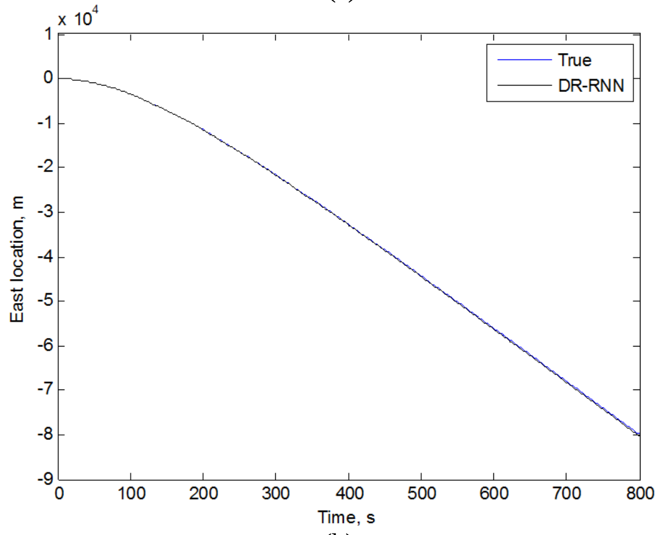
(b)

Figure 5. True and predicted responses of the aircraft using trained DR-RNNs for: (a) Case 2; (b) Case 3.

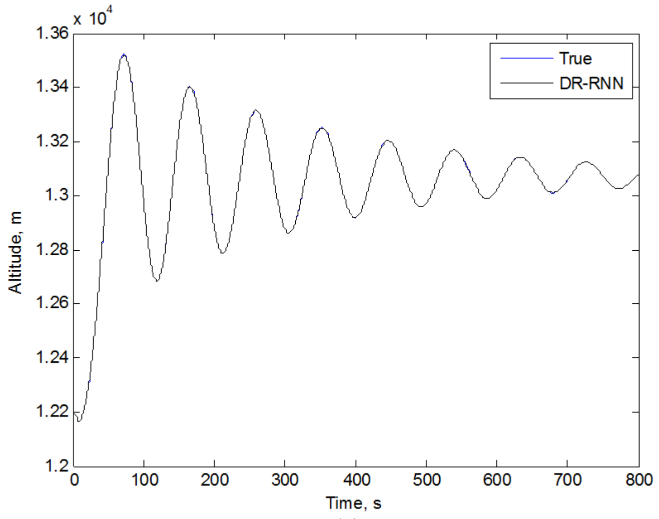
For better visualization, the predicted flight trajectory of Cases 5 is calculated using Eq. (12) and plotted in Figure 6. The accuracy of prediction is confirmed by the good match between the simulated and true flight trajectories. In addition, in order to test the training robustness of the DR-RNN, the training was conducted with the Gaussian white noise added to the training data where the signal to noise ratio of the training data is set as 50. Figure 7 shows the comparison between the predicted longitudinal responses and training data using the trained network. It can be seen that while there exists some errors in the predicted responses, the DR-RNN is still able to predict the general trend of the responses with sufficient accuracy.



(a)



(b)



(c)

Figure 6. True and simulated flight trajectories using trained DR-RNNs for Case 5: (a) North location; (b) East location; (c) Altitude

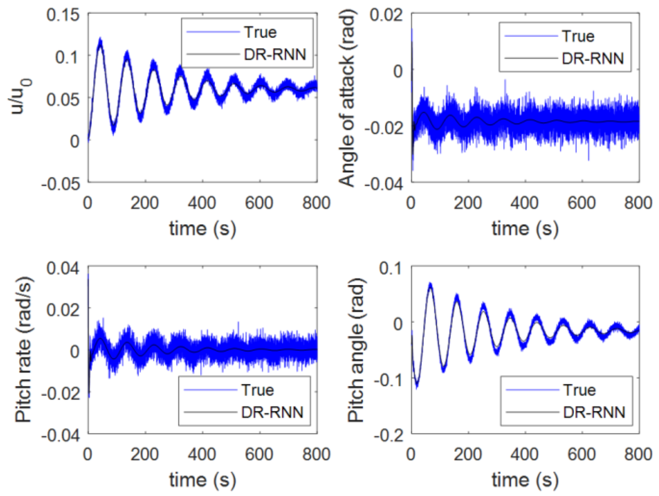
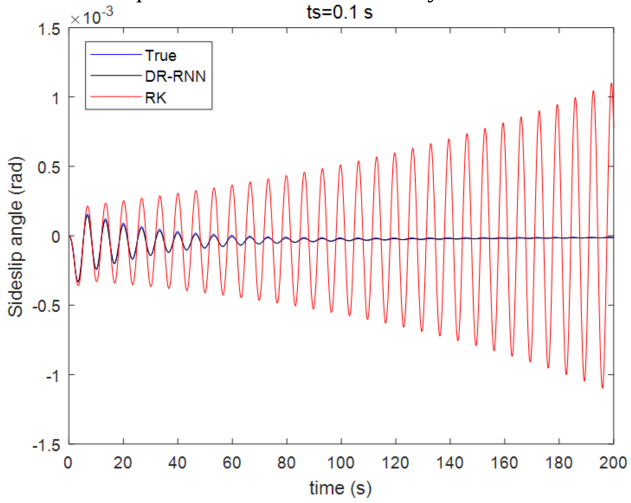


Figure 7. Prediction results using network trained with noisy data

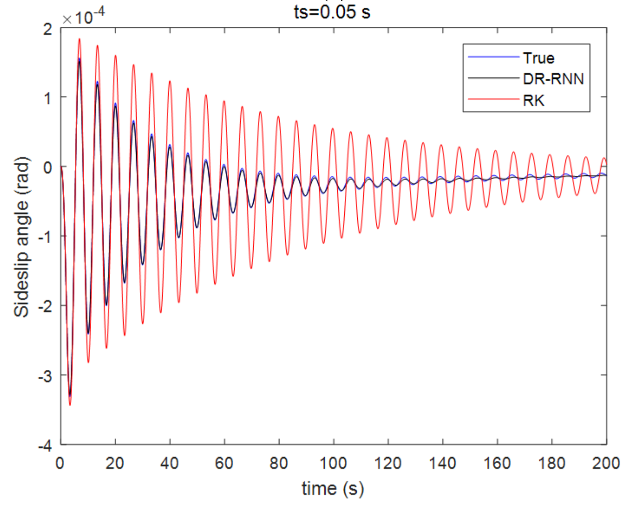
4.4. Computation Efficiency

Computation efficiency is a critical index to evaluate the performances of surrogate models. In this study, the computation efficiency of the DR-RNN is compared with that of a classical numerical method, the fourth-order Runge-kutta (RK) method. For both methods, the responses are obtained using two different time step sizes of 0.05 s and 0.1 s. Figure 8 plots the responses of the sideslip angle of the aircraft obtained using the DR-RNN and the RK method for Case 4 along with the true response. It can be seen that for both step sizes, the DR-RNN is able to accurately predict the time history of the sideslip angle. However, the responses obtained using the RK method showed much larger errors than those predicted using the DR-RNN. This is caused by the numerical instability of the RK method. In fact, Figure 8 shows that the response obtained using the

RK method tends to gradually converge to the true response with smaller step sizes. Nevertheless, for explicit method, a very small time step size is sometimes needed in order to achieve numerical stability. In practice, a small step size would considerably reduce the computation efficiency. From this perspective, the DR-RNN can efficiently reduce the simulation costs since it is able to use time scales larger than those required for numerical stability.



(a)



(b)

Figure 8. Time histories of sideslip angle for different time step sizes obtained using the DR-RNN and the RK method: (a) step size = 0.1 s; (b) step size = 0.05 s

In order to better demonstrate the computation efficiency of the DR-RNN, one hundred simulations of Case 4 with random initial disturbances to the aircraft states were conducted on a machine with Intel i7-3770 CPU and 16 GB of RAM using both the DR-RNN and the RK method. Table 6 lists the computation time for both methods as well as the average prediction error. The step size of the RK method is chosen such that its prediction error is comparable with that of the DR-RNN. From Table 5, it can be seen that the prediction using the DR-RNN is about 80 times faster than that of the RK method with a step size of 0.002 s and about 30 times faster than that of the RK method with a step size of 0.005 s. Also, it can be seen that the DR-RNN has the least prediction error among the three cases. Therefore, the DR-RNN is considered to be suitable for the surrogate modeling of aircraft dynamical systems for its superior computation efficiency.

Table 6. Comparison of computation time and accuracy between the DR-RNN and the RK method.

	DR-RNN (step size = 0.1 s)	RK (step size = 0.002 s)	RK (step size = 0.005 s)
Computation time (s)	7.4	605.4	241.1
Average prediction error	2.60e-4	3.78e-4	5.72e-4

5. CONCLUSIONS

In this study, the concept of physics-based learning is proposed and introduced to simulate aircraft dynamics. A physics-aware RNN known as the deep residual RNN (DR-RNN) is adopted to learn the aircraft dynamical behavior. The trained DR-RNN was used to perform predictions of aircraft responses under arbitrary control inputs and state disturbances. The prediction results show that the DR-RNN has excellent extrapolation capabilities in that: (1) it can accurately predict responses of much longer duration than that of the training data; (2) it can accurately predict responses under control inputs and disturbances different from those given during training. The excellent extrapolation capabilities of DR-RNN are attributed to the integration of the underlying physics of dynamical systems into the learning model. Also, it was discovered that the DR-RNN is robust in training since it is able to learn the aircraft dynamical behavior using training data contaminated with noise.

Furthermore, the DR-RNN demonstrates superior computation efficiency compared with a classical numerical method, the fourth-order Runge-kutta (RK) method. The main reason is that the DR-RNN is able to predict the aircraft responses using relatively larger time scales that violate the numerical stability conditions. Also, since the DR-RNN is explicit in time, the computational cost at each time step is fixed. Therefore, the DR-RNN is considered to be suitable for the surrogate modeling of dynamical systems. In addition, since the gradient information can be explicitly obtained through the training process, physics-based learning can be potentially applied for system identifications of dynamical systems, which will be investigated in future studies for the diagnostics and prognostics of dynamical systems.

It should be noted that in this study, the learned model is a linearized model of the non-linear differential equations, made at an initial cruise condition. Future study will focus on simulating the entire flight envelop by using several learned models corresponding to different stages of the flight. In addition, the performances between traditional RNNs (without physics) and DR-RNN will be compared in the future work.

ACKNOWLEDGEMENT

The research reported in this paper was supported by funds from NASA University Leadership Initiative program (Contract No. NNX17AJ86A, PI: Yongming Liu, Project Officer: Kai Goebel). The support is gratefully acknowledged.

REFERENCES

Ba, J.L., Kiros, J.R., & Hinton, G.E. (2016). Layer Normalization. *arXiv preprint*, arXiv:1607.06450 [stat.ML].

Brizt, D. (2015) *Recurrent Neural Networks Tutorial, Part 1: Introduction to RNNs*. http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Cho, K., van Merrienboer, B., Gulcehre, C., et al. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv preprint*, arXiv:1406.1078 [cs.CL].

Etkin, B, & Reid, L. (1996) *Dynamics of Flight Stability and Control*. Hoboken, NJ: John Wiley and Sons, Inc.

Goodfellow, I, Bengio, Y, and Courville, A. (2016). *Deep Learning*. Cambridge, MA: MIT Press.

Graves, A. (2013) Generating sequences with recurrent neural networks. *arXiv preprint*, arXiv:1308.0850.

Hinton, G., Deng, L., Yu, D., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, vol. 29(6), pp. 82-97.

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, vol. 9(8), pp. 1735-1780.

Kani, J.N., & Elsheikh, Ahmed. (2017). DR-RNN: A deep residual recurrent neural network for model reduction. *arXiv preprint*, arXiv:1709.00939.

Kingma D.P. & Ba, J. (2014) Adam: A Method for Stochastic Optimization, *3rd International Conference for Learning Representations*, May 7-9, 2015, San Diego, CA.

Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. *Proceedings of the 30th International Conference on International Conference on Machine Learning*, vol. 28, pp. 1310-1318.

Pete, M.M. (2010). Lecture 9: 6-DOF Equations of Motion. *MAE441: Spacecraft and Aircraft Dynamics*.

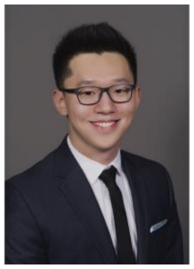
Tieleman, T., & Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4(2).

Trischler, & A.P., D'Eleuterio, G. (2016). Synthesis of recurrent neural networks for dynamical system simulation. *Neural Networks*, vol.80, pp. 67-78.

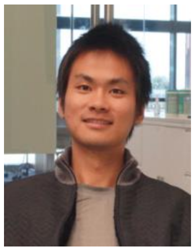
Werbos, P.J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, vol. 78(10), pp. 1550-1560.

Wikepedia. (2018, 05 10). *Boeing 747*. Retrieved from WIKEPEDIA: The Free Encyclopedia: https://en.wikipedia.org/wiki/Boeing_747

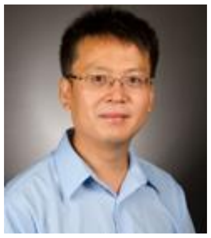
BIOGRAPHIES



Yang Yu is a postdoctoral research associate in the School for Engineering of Matter, Transport & Energy at Arizona State University. He received his Ph.D. in Civil Engineering in 2017 from Louisiana State University and his Bachelor's degree in Civil Engineering in 2014 from Hunan University, China. His research interests include applications of machine learning, structural health monitoring, structural safety and reliability, and structural dynamics.



Houpu Yao is currently a Ph.D. candidate in Aerospace Engineering at Arizona State University, Tempe, Arizona. He received his bachelor degree from Northwestern Polytechnical University, Xi'an, Shanxi, in 2013. His previous research interests include structural dynamics, computational mechanics and acoustics, boundary element methods and fast multipole methods, parallel computation, deep learning and computer vision. His current research interest is in bridging the gap between deep learning and computational mechanics, designing neural networks to solve various physics problems.



Yongming Liu is a professor in the School for Engineering of Matter, Transport & Energy at Arizona State University. He completed his PhD at Vanderbilt University in 2006, and obtained his Bachelors' and Masters' degrees from Tongji University in China in 1999 and 2002, respectively. His research interests include probabilistic methods, diagnostics and prognostics, risk management, materials and structures. He has published over 100 journal articles in the general area of prognostics and health management. He has served on many technical committees in AIAA, ASME, and ASCE. He is an associate fellow of AIAA.