Soft Computing (2022) 26:6907–6924
https://doi.org/10.1007/s00500-021-06639-8

APPLICATION OF SOFT COMPUTING

Feature extraction-based intelligent algorithm framework with neural network for solving conditional nonlinear optimal perturbation

Shijin Yuan[1] · Huazhen Zhang[1] · Yaxuan Liu[1] · Bin Mu[1]

Accepted: 30 November 2021 / Published online: 29 January 2022

Abstract

Conditional nonlinear optimal perturbation (CNOP) defines an optimization problem to study predictability and sensitivity of the oceanic and climatic events in the nonlinear system. One effective method to solve the corresponding problem is feature extraction-based intelligent algorithm (FEIA) framework. In the previous study, the mapper and the re-constructor of the framework are generally obtained by principal component analysis (PCA), but the solving performance still needs to further improve. Recently, neural network has attracted the attention of lots of researcher, and many structures of neural network can be used to construct the mapping-reconstruction structure of FEIA framework. However, the related studies applying neural network in FEIA framework are lacking. Compared with PCA, neural network might obtain a proper structure for FEIA framework with the well-directed training. Therefore, this paper suggests two ways applying neural network in FEIA framework, and the corresponding frameworks are tested to solve CNOP of double-gyre variation in Regional Ocean Modeling System (ROMS). The results show that FEIA framework with neural network can obtain the solutions with better objective function values, and the corresponding solutions have a larger probability leading to the related physical phenomenon.

Keywords Neural network · Intelligent algorithm · Large-scale optimization · Dimension reduction · Conditional nonlinear optimal perturbation

1 Introduction

Conditional nonlinear optimal perturbation (CNOP) method is proposed by Mu et al. (2003) to study predictability and sensitivity of the oceanic and climatic events in the nonlinear system (Wang et al. 2009). CNOP defines a perturbation which could lead to the largest nonlinear development at the prediction time under the given constraints. The study of the solved perturbation could help the researchers understand the corresponding physical mechanism and improve the prediction skill. Therefore, CNOP has been widely applied in the studies of the various events such as El Niño–Southern Oscillation (ENSO) (Zhang et al. 2018), typhoon (Mu et al. 2019) and Kuroshio (Zhang et al. 2019a, b, c), and these studies verify the efficiency of CNOP.

One problem in studying CNOP is how to obtain the perturbation leading to the largest development. Because the essence of CNOP is an optimal problem, there are two general ways to solve it. One way is applying the gradient-based methods (Sun et al. 2010), but these methods are easy to encounter the problems such as the missing of the adjoint component and the influence of the discontinuous on–off switch (Mu et al. 2005). Another way is applying the gradient-free methods, for example, intelligent algorithms, also called heuristic algorithm, could be applied to solve CNOP (Zheng et al. 2014). However, intelligent algorithm generally cannot obtain an effective solution within a proper time when the scale of the problem is large.

✉ Bin Mu
binmu@tongji.edu.cnss

Shijin Yuan
yuanshijin2003@163.com

Huazhen Zhang
yikui691827@qq.com

Yaxuan Liu
yaxuan_liu@qq.com

[1] School of Software Engineering, Tongji University, Shanghai 201804, China

Therefore, some researchers propose a framework, which is called feature extraction-based intelligent algorithm (FEIA) framework in this paper, to reduce the search space of intelligent algorithm by the dimension reduction method such as principal component analysis (PCA) (Wold et al. 1987; Ringnér 2008; Abdi and Williams 2010). For example, Mu et al. (2015) combine the PCA and particle swarm optimization (PSO) to solve the CNOP of ENSO in Zebiak–Cane (ZC) model and Yuan et al. (2019a) combine the PCA and simulated annealing (SA) to solve the CNOP of double-gyre variation in Regional Ocean Modeling System (ROMS). Although utilizing PCA improves time efficiency to some extent, there exists a problem that due to the fixed latent space in PCA, the probability of obtaining the effective solution is quite low.

Recently, neural network has attracted the attention of lots of researchers because of its convenience and excellent performance. Meanwhile, many structures of neural network can be used to construct the low-dimensional latent search space. Compared with PCA, neural network might obtain a relative sparse uniform search space or a better reconstruction mapping for the special problem, which might be helpful for searching. However, few studies concentrate on employing neural network as feature extraction component in FEIA framework to elevate the solving performance. Hence, in this paper, two possible ways applying neural network in FEIA framework are thought of. One way is trying applying the neural network with the reduction function. For example, auto-encoder (AE) (Rumelhart et al. 1986; Hinton et al. 2006; Wang et al. 2016; Ramamurthy et al. 2020) and its variants such as sparse AE (SAE) (Ng 2011; Liu et al. 2019; Zhang et al. 2021), convolutional AE (CAE) (Masci et al. 2011; Chen et al. 2018) and variational AE (VAE) (Kingma et al. 2014; Xie et al. 2019; Liu et al. 2020a, b; Lin et al. 2020; Jiao et al. 2020) might replace the role of PCA in FEIA framework. Another way is applying PCA to obtain the latent space and applying neural network, such as decoder and generative adversarial nets (GAN) (Goodfellow et al. 2014; Creswell et al. 2018; Zhang et al. 2019a, b; Schonfeld et al. 2020), to reconstruct the origin space. Then, we conduct experiments to verify the feasibility of adopting the above two ways to solve the CNOP of double-gyre variation in ROMS. Results demonstrate that in contrast to PCA-based FEIA, FEIA with neural network can obtain more effective solutions in the aspects of better objective values and larger probabilities triggering the expected physical phenomenon.

The rest of this paper is organized as follows. This paper selects solving CNOP of double-gyre variation in ROMS as the case study, and the related contents of CNOP, neural network-based dimension reduction methods and the case are described in Sect. 2. In Sect. 3, FEIA framework whose intelligent algorithm component is selected as PSO, the related neural network and the coupling way of the network for FEIA framework are introduced. The tuning process of the network and the solving result are shown and analyzed in Sect. 4. Finally, the conclusion and the prospect of future work are given in Sect. 5.

2 Related works

2.1 CNOP

CNOP method, proposed by Mu et al. (2003), has been widely applied in the field of atmospheric and oceanic sciences to study the predictability and sensitivity (Wang et al. 2020; Jiang and Duan 2020; Liu et al. 2020a, b).

Mathematical description of CNOP is mentioned in the following.

For a given problem, assume that $\boldsymbol{X_0}$ is the initial state, M_t is the nonlinear propagator of the model from time 0 to t and $\boldsymbol{x_0}$ is the initial perturbation. In order to explore the initial perturbation $\boldsymbol{x_0^*}$, which can make the model's development deviate maximally from the reference state at the prediction time under the given constraints δ, the problem of CNOP can be written as follows (Eq. 1):

$$\begin{aligned} \mathbf{x_0^*} &= \arg \max_{\|\mathbf{x_0}\|_C \leq \delta} J(x_0) \\ &= \arg \max_{\|\mathbf{x_0}\|_C \leq \delta} \|M_t(\boldsymbol{X_0} + \mathbf{x_0}) - M_t(\mathbf{X_0})\|_E \end{aligned} \tag{1}$$

where $\|\cdot\|_E$ is the energy norm and $\|\cdot\|_C$ is the constraint norm for the problem.

The essence of CNOP is an optimal problem with certain constraints. Generally, depending on whether the gradient is involved, there are two types of approaches to solving CNOP. One is applying gradient-based methods (Sun et al. 2010). The adjoint method, a traditional gradient-based method, is highly dependent on adjoint component in numerical model and requires a lot of computation (Towara and Naumann 2013). Besides, due to frequent occurrence of discontinuous on–off switch (Mu et al. 2005) in nonlinear system, gradient-based methods fail to compute the correct gradient, which results in failure of solving CNOP. The other is applying gradient-free methods. Intelligent algorithms are the general name of a series of gradient-free algorithms designed by people utilizing natural laws. Such algorithms could be applied to solve CNOP (Zheng et al. 2014). When the scale of problem is small, intelligent algorithms could obtain global optimum. However, the scale of oceanic and climatic events is large. In this case, intelligent algorithms cannot obtain an effective solution within a proper time. With the aim of reducing the search space of intelligent algorithms, FEIA framework is proposed. PCA (Wold et al. 1987; Ringnér 2008; Abdi and Williams 2010) is often used as the dimension reduction method in FEIA framework. For example, Mu et al. (2015) combine the PCA and particle swarm optimization (PSO) to solve the CNOP of ENSO in Zebiak–Cane (ZC) model and Yuan et al. (2019a) combine the PCA and simulated annealing (SA) to solve the CNOP of double-gyre variation in Regional Ocean Modeling System (ROMS). Although utilizing PCA improves time efficiency in some way, there exists a problem that the probability of obtaining the effective solution is quite low, since PCA with fixed latent space cannot balance the trade-off between dimension reduction and information loss in some cases.

2.2 Neural network-based dimension reduction

With the development of artificial intelligence, many structures of neural network can be used to construct a relative sparse uniform latent search space or a better reconstruction mapping. AE is a neural network structure dedicated to transforming inputs into outputs with minimal information loss. Rumerlhart et al. (1986) first introduced AE in 1980s. AE can map input data to low-dimensional latent space and then use that latent space to generate output data similar to the input data. Because of its ability of learning useful features from data, AE has been universally applied in dimension reduction. Wang et al. (2016) compare AE with state-of-the-art dimension reduction methods. Experimental results show that AE can learn some features that are different from other dimension reduction methods. Researchers also apply AE in hyperspectral images classification as the dimension reduction component and prove that this proposed technique achieves image denoising and high performance (Ramamurthy et al. 2020). Besides, there are some variants of AE such as SAE, CAE and VAE. Ng (2011) introduces the concept of sparsity into training of traditional autoencoders and proposes SAE to enable the hidden variables to show more obvious characteristics. SAE has been widely used in feature extraction of images (Liu et al. 2019; Zhang et al. 2021). By introducing convolutional layers and deconvolutional layers, CAE is proposed to enable the network to better capture spatial features and relevance between data. It is applied in anomaly detection to learn nonlinear relationships between features (Chen et al. 2018). VAE assumes the latent feature follows a distribution, which is generally Gaussian distribution. VAE-based dimension reduction is applied in text learning (Xie et al. 2019; Liu et al. 2020a, b), biology-related analysis (Lin et al. 2020; Jiao et al. 2020), etc. In addition, GAN, proposed by Goodfellow et al. (2014), shows high potentials in data generation (Creswell et al. 2018). GAN and its variants (Zhang et al. 2019a, b; Schonfeld et al. 2020) generally consist of generator to generate simulated data and discriminator to discriminate whether the date is real or not. The structure of GAN meets requirements of reconstruction.

2.3 The case of double-gyre variation in ROMS

Double gyre, which consists of a sub-polar gyre and a subtropical gyre, is a typical large-scale ocean circulation in the northern mid-latitude ocean basins (Shen et al. 1999). Double-gyre variation is one of the low-frequency variability phenomena (Nauw and Dijkstra 2001). The study of the variation is helpful to understand the dynamic mechanism of double gyre and how the oceanic variability contributes to the mid-latitude climate variability (Qiu 2000).

ROMS is a split-explicit, free-surface, topography-following-coordinate ocean model (Shchepetkin & McWilliams 2005). It has been widely used in a variety of applications of the scientific community. Double gyre is one of the events simulated in ROMS, and the simulation follows Moore et al. (2004). The model simulates double gyre in a region whose longitude length and latitude length are separately 1000 km and 2000 km, respectively, and the region is divided into four vertical layers of 125 m in the vertical direction. The state data of double gyre consist of three parts, which are separately eastward velocity $\boldsymbol{u}$, northward velocity $\boldsymbol{v}$ and sea surface height $\boldsymbol{\zeta}$. Under the resolution of 18.5 km, the size of $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{\zeta}$ is separately $55 \times 110 \times 4$, $56 \times 109 \times 4$ and 56×110, and the total size of the state data is 54776.

Generally, double gyre can be divided into three states, which are separately symmetry state (Fig. 1a), jet-up state (Fig. 1b) and jet-down state (Fig. 1c). Figure 1 shows the representation of the three states in ROMS. In general, double gyre keeps steady in one state or shifts between symmetry state and another state. When the variation happens, the shift between jet-up state and jet-down state would appear. CNOP can be used to obtain the initial perturbation causing the variation, and the obtained perturbation can be used to study the dynamic mechanism of double gyre (Yuan et al. 2019a).

According to Zhang et al. (2015), the energy norm (Eq. 2) and the constraint norm (Eq. 3) of double gyre can be defined as follows:

$$\begin{aligned}\|M_t(\mathbf{X}_0+\mathbf{x}_0)-M_t(\mathbf{X}_0)\|_E \\ =\frac{1}{2}\left[\int_\Lambda h\left(\Delta\mathbf{u}_\mathbf{t}^2+\Delta\mathbf{v}_\mathbf{t}^2\right)dxdydz+\int_\Lambda \Delta\zeta_\mathbf{t}^2 dxdy\right]\end{aligned} \tag{2}$$

$$\|\mathbf{x}_0\|_C=\frac{1}{2}\left[\int h\left(\Delta\mathbf{u}_\mathbf{t}^2+\Delta\mathbf{v}_\mathbf{t}^2\right)dxdydz+\int g\Delta\zeta_\mathbf{t}^2 dxdy\right] \tag{3}$$

where $\boldsymbol{X_0} = \{\boldsymbol{u_0},\ \boldsymbol{v_0},\ \boldsymbol{\zeta_0}\}$ and $\boldsymbol{x_0} = \{\boldsymbol{\Delta u},\ \boldsymbol{\Delta v},\ \boldsymbol{\Delta\zeta}\}$ are separately the initial state vector and the initial perturbation state vector for double-gyre data simulated in ROMS. M_t is the nonlinear propagator of ROMS from time 0 to t, which

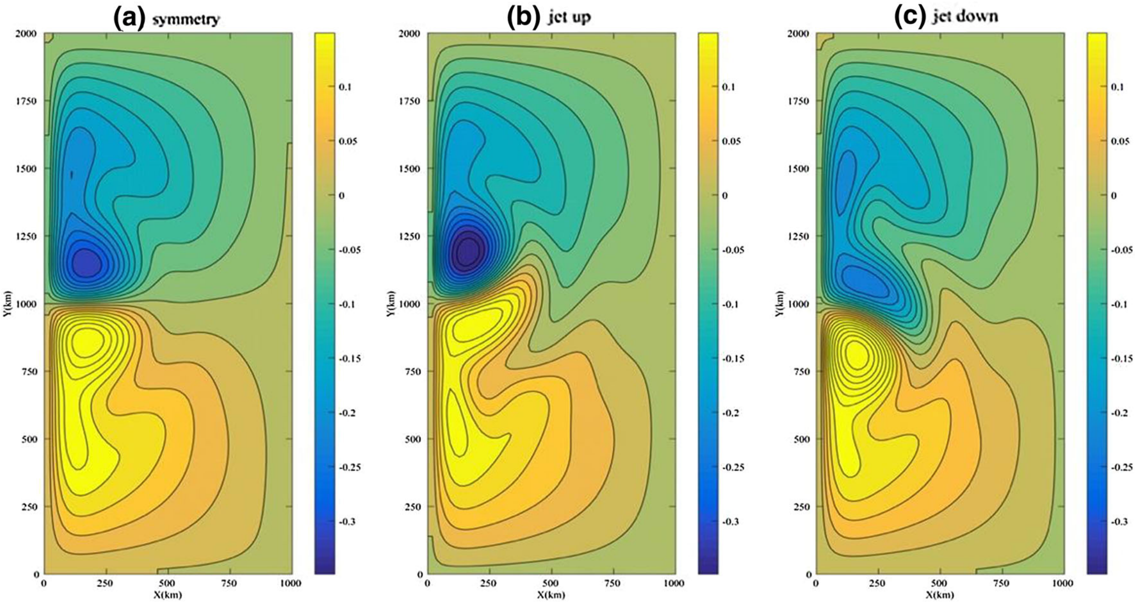


Fig. 1 The **a** symmetry state, **b** jet-up state, **c** jet-down state of double gyre

can be regarded as a black-box function. $\{\Delta\boldsymbol{u}_t, \Delta\boldsymbol{v}_t, \Delta\boldsymbol{\zeta}_t\}$ is the development state vector, which is calculated from the result of $M_{t,}$ and can be calculated as follows:

$$\begin{aligned}\{\Delta\mathbf{u_t}, \Delta\mathbf{v_t}, \Delta\zeta_\mathbf{t}\} &= \{\mathbf{u_{t\_new}}, \mathbf{v_{t\_new}}, \zeta_\mathbf{t\_new}\} - \{\mathbf{u_t}, \mathbf{v_t}, \zeta_\mathbf{t}\} \\ &= M_t(\mathbf{X_0} + \mathbf{x_0}) - M_t(\mathbf{X_0}).\end{aligned} \tag{4}$$

g is the gravitational acceleration, whose value is set to 9.8 m/s^2, and h is the vertical layer thickness, whose value is 125 m. The energy norm integrates in the region Λ $(0\text{ km} \le x \le 600\text{ km},\ 750\text{ km} \le y \le 1250\text{ km})$, and the constraint norm integrates in the whole simulation region $(0\text{ km} \le x \le 1000\text{ km},\ 0\text{ km} \le y \le 2000\text{ km})$. The other settings refer to the previous experiment (Yuan et al. 2019a), where the initial state $\boldsymbol{X_0}$ is a jet-up state and the constraint value of perturbation δ is set to 4.0 × 10^{11} m^5/s^2, which is about 10% of the constraint norm value of the initial state. The intent of the problem is to obtain the perturbation $\mathbf{x}_\mathbf{0}^*$,which can lead to double-gyre variation.

3 Methods

3.1 FEIA framework

Intelligent algorithm is a type of method combining rules and randomness to imitate the natural phenomena and seek the optimal value (Lee et al. 2005). The basic flows of intelligent algorithm can be summarized as follows:

Step 1: Determine the initial solution $\boldsymbol{x}$.

Step 2: Calculate the objective function value f with $\boldsymbol{x}$.

Step 3: Judge the iteration condition. If the termination condition is satisfied, output the best solution $\boldsymbol{x}^*$; otherwise, go to Step 4.

Step 4: Update solution $\boldsymbol{x}$ with the related rules and the objective value f calculated in Step 2. Go to Step 2.

One problem of intelligent algorithm is curse of dimensionality. Assume the scale of the problem, i.e., the dimension of $\boldsymbol{x}$, is n. Because the essence of intelligent algorithm is doing the random search with some rules in the n-dimension space, if n is too large, the efficiency of the algorithm would be very low. However, in the process of solving the actual problem, the solution generally has some features. Assume the whole n-dimension space is O, the points with the related features in O make up a subspace S. For example, in the CNOP problem of this paper, the optimal initial perturbation is in the subspace, which shows the perturbation feature of double gyre. If there are two mappings p and r, p can map x in S into an m-dimension $(m < < n)$ latent space F and r can reconstruct the low-dimensional solution $\boldsymbol{w}$ in F into $\boldsymbol{x}$. One optimal problem can be reconstructed as follows (Eq. 5):

$$\begin{aligned}\mathbf{x}^* = r(\mathbf{w}^*) = \arg optf(r(\mathbf{w})) \\ s.t.g_i(\mathbf{x}) = g_i(r(\mathbf{w})) \ge 0, i = 1, \ldots, m\end{aligned} \tag{5}$$

where f is the objective function and g_i is the constraint.

Based on the above thought, this paper calls the process that intelligent algorithm solves the optimal problem in the low-dimensional space as FEIA framework. The basic flow of FEIA framework can be summarized as follows:

Step 1: Collect the samples in F. Determine the mapper p and the re-constructor r.

Step 2: Determine the initial solution $\boldsymbol{x}$. Map x into $\boldsymbol{w}$ by p.
Step 3: Reconstruct $\mathbf{w}$ into $\boldsymbol{x}$ by r. Calculate the objective function value f with $\boldsymbol{x}$.
Step 4: Judge the iteration condition. If the termination condition is satisfied, output the best solution $\boldsymbol{x}^* = r(\boldsymbol{w}^*)$; otherwise, go to Step 4.
Step 5: Update solution $\boldsymbol{w}$ with the related rules and the objective value f calculated in Step 3. Go to Step 3.

One point is the initial solution in Step 2 of FEIA framework should also in F, so the construction of the initial solution would combine experience and randomness rather than completely relies on randomness. In past, the mapper and re-constructor usually are the feature matrix and its transpose matrix of PCA. And in this paper, PSO algorithm is selected the intelligent algorithm component of FEIA framework. Therefore, PCA and PSO algorithms are introduced briefly as follows.

3.1.1 PCA

PCA is a classical machine learning method, which is widely applied in the dimension reduction problem. The intent of PCA is to obtain a set of vector bases making the data have the maximum projection in the direction of the vector basis, so the approximate error between the reduction data and the original data could be as small as possible. Assume $\boldsymbol{X} \subseteq F$ is the sample set of the possible solutions. The vector bases can be obtained by doing eigen-decomposition for $\boldsymbol{X}$ (Eq. 6).

$$\mathbf{U}, \Sigma = eigen\_decom(\mathbf{X}\mathbf{X}^{\mathbf{T}}) \tag{6}$$

where $\boldsymbol{\Sigma} = \{\lambda_1, \ \lambda_2, \ldots, \ \lambda_n\}$ is the eigenvalues of the descending order and $\boldsymbol{U} = (\boldsymbol{u_1}, \ \boldsymbol{u_2}, \cdots, \ \boldsymbol{u_n})$ is the corresponding eigenvectors. Assuming n is the dimension of the original data and m is the dimension of the reduction data, the vector bases consist of the first m eigenvectors $\boldsymbol{U_m} = (\boldsymbol{u_1}, \ \boldsymbol{u_2}, \cdots, \ \boldsymbol{u_m})$. If PCA is the feature extraction component of FEIA framework, p and r can be constructed as $\boldsymbol{U_m}$ (Eq. 7):

$$p(\mathbf{x}) = \mathbf{x}\mathbf{U_m}, r(\mathbf{w}) = \mathbf{w}\mathbf{U}_{\mathbf{m}}^{T}. \tag{7}$$

3.1.2 PSO

PSO algorithm is an intelligent algorithm imitating the process that birds search for food. The algorithm assumes there are l particles searching the optimal solution and updates the particles by recording the local best solution and global best solution. The local best solution is the best solution for each particle in the past iterations, and the global best solution is the best solution for all the particles in the past iterations. For FEIA framework, the pseudocode

```
Pseudocode: PSO component for FEIA framework (For the maximum problem)
Parameter: the initial solution $\boldsymbol{x_0}$, the particles' number $l$, the max iteration time $max\_iter$, the initial inertia coefficient $i_{c0}$, the decrement of inertia coefficient $\Delta i_c$, the learning factors $c_1$ and $c_2$, the mapper $p$ and the re-constructor $r$
$\boldsymbol{w_0} = p(x_0)$
$i_c = i_{c0}$
for $j = 1 : l$
  randomly generate m-dimension velocity vector $v^j$
  $w^j = w_0 + v^j$
for $i = 1: max\_iter$
  for $j = 1 : l$
    $\boldsymbol{x^j} = r(\boldsymbol{w^j})$
    if $f(\boldsymbol{x^j}) > f(r(\boldsymbol{w^{j\_local}}))$
      $\boldsymbol{w^{j\_local}} = \boldsymbol{w^j}$
      if $f(\boldsymbol{x^j}) > f(r(\boldsymbol{w^{global}}))$
      $\boldsymbol{w^{global}} = \boldsymbol{w^j}$
  for $j = 1 : l$
    $\boldsymbol{v^j} = i_c \cdot \boldsymbol{v^j} + c_1 \cdot rand(0,1) \cdot (\boldsymbol{w^{j\_local}} - \boldsymbol{w^j}) + c_2 \cdot rand(0,1) \cdot (\boldsymbol{w^{global}} - \boldsymbol{w^j})$
    $\boldsymbol{w^j} = \boldsymbol{w^j} + \boldsymbol{v^j}$
    $\boldsymbol{w_j} = cons\_pro\ (\boldsymbol{w_j})$
  $i_c = i_c - \Delta i_c$
Output the best solution $r(\boldsymbol{w^{global}})$
```

of PSO algorithm component is as follows.

In the pseudocode, the initial solution $\boldsymbol{x_0}$ is an empirical solution which follows the feature of the space F. The inertia coefficient i_c is the adaptive parameter to keep the past velocity. In this paper, most constants of the algorithm are set to the empirical values, where l is set to 20, i_{c0} is set to 0.9, Δi_c is set to 0.01 and c_1 and c_2 are set to 2. Because the calculation of the model costs lots of time in the CNOP problem, *max_iter* is set to 30 for verifying whether the algorithm can obtain the effective solution in the proper time. One important thing to note is that a constraint project function should be defined if the problem has some constraint. For the CNOP problem in this paper, the project function is defined as Eq. 8.

$$cons\_pro(\mathbf{w^j}) = \begin{cases} \mathbf{w^j}, & \text{if}||r(\mathbf{w^j})||_C \leq \delta \\ \sqrt{\dfrac{\delta}{||r(\mathbf{w^j})||_C}}\mathbf{w^j}, & \text{while}||r(\mathbf{w^j})||_C \end{cases}. \tag{8}$$

3.2 AE and its variants

AE is a common neural network for dimension reduction. Compared with other neural network, AE has the following characteristics: (1) the number of units in the input layer and that in the output layer are same; (2) the number of the units in the middle latent layer is less than that in input layer or output layer; (3) the network from the input layer to the middle latent layer is called encoder and the network from the middle latent layer to the output layer is called decoder; and (4) the training for the network should make the input and output as same as possible. Figure 2 shows a simple AE with three layers.

In Fig. 2, l_1 is the input layer, l_2 is the middle latent layer and l_3 is the output layer. l_1 and l_2 make up the encoder, and l_2 and l_3 make up the decoder. The relation between two adjacent data layers can be written as follows (Eq. 9):

$$\mathbf{X_{i+1}} = h_i(\mathbf{X_i}|\mathbf{W}_i, \mathbf{b_i}) \tag{9}$$

where $\boldsymbol{X_i}$ is the data in i_{th} layer, h_i is a hidden function and $\boldsymbol{W_i}$ and $\boldsymbol{b_i}$ are the parameters of the function. It is easy to find that encoder and decoder of AE can serve as the mapper p and the re-constructor r in FEIA framework. In this paper, four kinds of AEs are tested in the experiment, and the follows introduce them briefly. And the detailed setting such as activation function is discussed in Sect. 3.4.

3.2.1 AE

For the original AE, the data vectors are passed in the network by the fully connected layer. Besides the three necessary data layers, there can be other latent data layers in the encoder and decoder. The hidden function can be written as follows (Eq. 10):

$$h_i(\mathbf{X_i}|\mathbf{W}_i, \mathbf{b_i}) = act_i(\mathbf{X_i}\mathbf{W}_i + \mathbf{b_i}) \tag{10}$$

where $act_i(\cdot)$ is the activation function, which is discussed in Sect. 3.4. $\boldsymbol{W_i}$ is the weight matrix in the fully connected layer, and $\boldsymbol{b_i}$ is the biases vector in the fully connected layer. The intent of AE is to minimize the reconstruction error, so the cost function of the network is defined as Eq. 11.

$$c(\mathbf{W}, \mathbf{b}) = \frac{1}{2}mse(\mathbf{X}, \mathbf{Y}) + reg(\mathbf{W}) \tag{11}$$

where $\boldsymbol{W}$ and $\boldsymbol{b}$ are the all weights and biases adjusted in the network, $\boldsymbol{X}$ is the input data, $\boldsymbol{Y}$ is the output data, $mse(\cdot)$

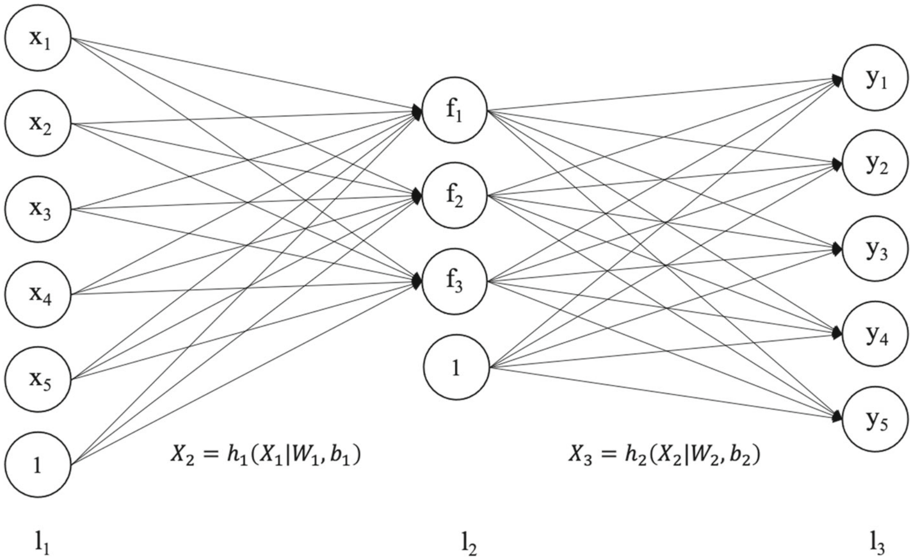


Fig. 2 A simple three-layer AE

(Eq. 12) represents the reconstruction error, and *reg(·)* (Eq. 13) is the regularization function to avoid overfitting.

$$mse(\mathbf{X},\mathbf{Y}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \tag{12}$$

$$reg(\mathbf{W}) = \lambda_{reg}\sum\nolimits_{i,j,l} w_{i,j,l}^2 \tag{13}$$

where λ_{reg} is a constant which represents the weight of the regularization cost and *i*, *j* and *l* are the indices of row, column and layer for the weight matrix, respectively.

3.2.2 SAE

The structure of SAE is similar to AE, and there are two main differences: (1) The data layers only contain the main three layers and (2) a sparse cost (Eq. 14) is added to the cost function (Eq. 15).

$$\begin{aligned} spa(\hat{\rho}) &= \lambda_{spa}\sum_{j=1}^{m} KL(\rho \parallel \hat{\rho}_j) \\ &= \lambda_{spa}\sum_{j=1}^{m} \rho\log\frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{\hat{\rho}_j} \end{aligned} \tag{14}$$

$$c(\mathbf{W},\mathbf{b}) = \frac{1}{2}mse(\mathbf{X},\mathbf{Y}) + reg(\mathbf{W}) + spa(\hat{\rho}) \tag{15}$$

where the sparse cost is represented by the Kullback–Leibler divergence between the activation degree of the middle layer data $\overset{\wedge}{\rho}$ and a sparseness constant ρ which is set to an experience value 0.1 in this paper. λ_{spa} is a constant which represents the weight of the sparse cost. By controlling the activation degree of the middle layer, SAE could relatively keep the sparseness of the feature.

3.2.3 CAE

The difference between CAE and AE is that CAE introduces the convolutional layer and the deconvolutional layer to replace the fully connected layer. Different from the fully connected layer, the passed data in the convolutional layer are matrix, and the hidden function can be written as Eq. 16:

$$h_i(\mathbf{X_i}|\mathbf{W}_i,\mathbf{b_i}) = act_i(p_{c\_d}(\mathbf{X_i},\mathbf{W_i}) + \mathbf{b_i}) \tag{16}$$

where P_{c_d} represents the process of the convolution (Fig. 3a) or deconvolution (Fig. 3b), $\boldsymbol{W_i}$ is the set of (de)-convolutional kernel and $\boldsymbol{b_i}$ is the biases matrix. As shown in Fig. 3a, the convolution is mapping the dot product of the shadow matrix in original data and kernel into the dot in the projection data. And as shown in Fig. 3b, the deconvolution is making the dot in original data multiply the kernel and then adding the result to the shadow matrix in the projection data. The cost function of CAE is same as that of AE (Eq. 11). Compared with AE, CAE can catch the spatial information of the data and the convolutional kernel can reduce the memory usage.

3.2.4 VAE

VAE assumes the latent feature follows a distribution, which is generally Gaussian distribution. Based on the assumption, the encoder of the network outputs a mean and a standard deviation of the distribution to construct the latent feature rather than outputs the latent feature directly, and Fig. 4 shows this flow. Meanwhile, the distribution cost (Eq. 17) is added to the cost function (Eq. 18).

$$dis(\boldsymbol{\mu},\boldsymbol{\sigma}) = \frac{\lambda_{vae}}{2}\sum_{i=1}^{m} 1 + \log(\sigma_j^2) - \mu_i^2 - \sigma_i^2 \tag{17}$$

$$c(\mathbf{W},\mathbf{b}) = \frac{1}{2}mse(\mathbf{X},\mathbf{Y}) + reg(\mathbf{W}) + dis(\boldsymbol{\mu},\boldsymbol{\sigma}) \tag{18}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are separately the mean vector and the standard deviation vector outputted by the encoder. λ_{vae} is a constant which represents the weight of the distribution cost.

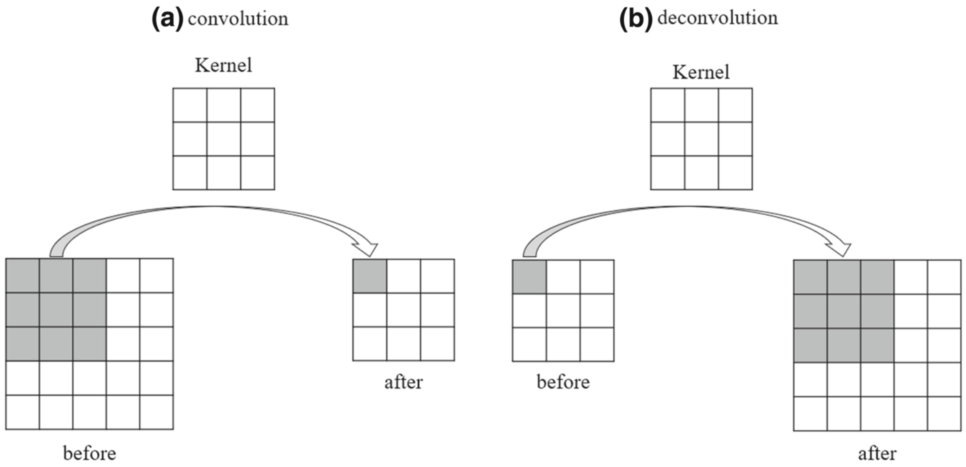


Fig. 3 The process of **a** convolution and **b** deconvolution

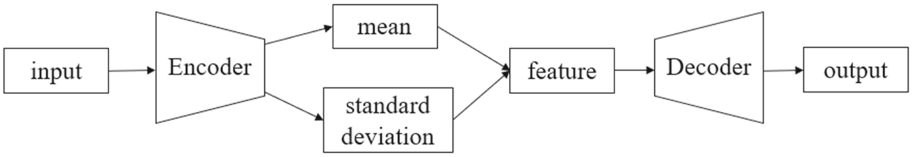


Fig. 4 The simple flow of VAE

3.3 The mapping model-based PCA and neural network

Another possible way applying neural network in FEIA framework is only training neural network as the re-constructor. This way assumes the latent features obtained by PCA are good enough and tries training some different re-constructor by the neural network. Compared with the neural network in Sect. 3.2, the input data $\boldsymbol{X}_w$ are the calculated by the original data $\boldsymbol{X}$ cross-multiples the reduction matrix $\boldsymbol{U}_m$ obtained by PCA and the output data $\boldsymbol{Y}$ are expected to be the same as the original data $\boldsymbol{X}$. In this paper, the decoder and GAN are tested as the re-constructor of this way in the experiment, and the follows introduce them briefly.

3.3.1 Decoder

As the name indicates, the decoder is the second part of AE. The structure of the decoder can follow the description in Sect. 3.2.

3.3.2 GAN

GAN consist of a generator network G and a discriminator network D. The generator maps the latent data $\boldsymbol{X}_w$ into the real data space, and the discriminator evaluates and distinguishes the real data $\boldsymbol{X}$ and the generated data $\boldsymbol{Y}$. Figure 5 shows the basic structure of GAN.

It is clear that the generator of GAN can serve as the re-constructor in FEIA framework. The inner structure of the generator could be same as the decoder, and the difference is that the cost function (Eq. 19) of the generator consists of the cost function of the decoder and a generator discrimination cost (Eq. 20).

$$c_g(\mathbf{W}, \mathbf{b}) = \frac{1}{2} mse(\mathbf{X}, \mathbf{Y}) + reg(\mathbf{W}) + gan_g(\mathbf{Y}) \tag{19}$$

$$gan_g(\mathbf{Y}) = -\lambda_{\text{gan}} \log(D(\mathbf{Y})) \tag{20}$$

where $D(\cdot)$ represents the output of the discriminator and λ_{gan} is a weight constant for the cost of GAN. The inner structure can be a classifier network, whose dimension of output is 1 and activation function of output layer can be a sigmoid function (Eq. 21). The larger output of the discriminator represents that the input has a larger possibility to be real data. And the cost function (Eq. 22) of discriminator consists of a discriminator discrimination cost (Eq. 23) and a regularization cost.

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{21}$$

$$c_d(\mathbf{W}, \mathbf{b}) = reg(\mathbf{W}) + gan_d(\mathbf{X}, \mathbf{Y}) \tag{22}$$

$$gan_d(\mathbf{X}, \mathbf{Y}) = -\lambda_{gan}(\log(D(\mathbf{X})) + \log(1 - D(\mathbf{Y}))). \tag{23}$$

3.4 The coupling of neural network for FEIA framework

In this section, the settings of neural network for FEIA framework in the experiment are discussed. In detail, five points for the network are introduced in the following.

3.4.1 Activation function

The data for the double gyre in ROMS could be negative, and the reduction solution in FEIA framework also could be negative. Therefore, the activation function of the output layer for the mapper and the re-constructor is set to linear activation function. And the activation function of other latent layers is setting to leaky RELU function (Eq. 24).

$$leaky\_relu(x) = \begin{cases} x, & \text{if}\, x \geq 0 \\ ax, & \text{if}\, x < 0 \end{cases} \tag{24}$$

where a is the negative gradient factor which is set to the empirical value 0.2 in the experiment.

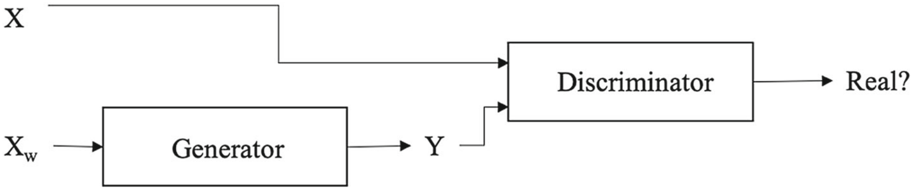


Fig. 5 The basic structure of GAN

3.4.2 Re-constructor bias

CNOP is an optimization problem with the constraint. It can be found that the bias could make the solution not satisfy the constraint. According to Eq. 8, although the project function changes the value of the latent solution w^j, the result of the norm is always influenced by b. For example, Eq. 25 shows the influence of b for computation of 2-norm in a two-layer linear network.

$$\|w_s(\tau x)+b\|_2 = (w_s(\tau x))^2 + 2w_s(\tau x) + b^2 \tag{25}$$

where w_s is the sum weight for the element x in data vector, b is the bias for x and τ is the project coefficient for x. It could be seen that τ does not influence b^2 in the result and the constraint could be never satisfied. Therefore, in the experiment, the bias for re-constructor is always set to 0, so the constraint could be satisfied with the constraint project function.

3.4.3 Weight parameter selection

As described in Sects. 3.2 and 3.3, many weight constants λ are introduced to construct the cost function of the network. Considering that the mean square error is the main cost, this paper solves the various costs in Sects. 3.2 and 3.3 without training, and the various λs are used to make the value of other cost be about one percent of the initial value of the mean square error. Therefore, the network cannot ignore the main intent and assisted by the other cost in the later training stage. The values of the λs are as follows: λ_{reg} is set to 10^{-6}, λ_{spa} is set to 10^{-3}, λ_{vae} is set to 10^{-4} and λ_{gan} is set to 10^{-4}.

3.4.4 Training data and validation data

The application of determining training data and validation data can help the training of the network. In FEIA framework, the samples in the possible solution space can be chosen as the training data, and the initial solution for the intelligent algorithm can be chosen as the validation data. Therefore, the fit degree of the network for the solving can be evaluated simply, and the network can be adjusted further.

For example, for the double gyre simulated in ROMS, a set of non-period oscillation data and a set of steady data are obtained by adjusting the model parameters according to Yuan (2019b). The difference between the non-period oscillation data and the steady data makes up the training data, whose size is 2000 × 54,776. And the training data are also the original matrix to carry out the process of PCA. On the other hand, the initial solution is constructed by the difference between jet-down data and symmetry data. And this initial solution for the intelligent algorithm component is the validation data.

3.4.5 Training process

The Adam optimizer (Kingma et al. 2014) is used to training the network in the experiment. And the flow of the training is as follows:

```
Pseudocode: Training Process of the network
Parameter: the training data set $X_{train}$, the validation data set $X_{val}$, the training epoch $epoch$, the training batch size $Size_{batch}$ and batch amount $cnt_{batch}$, the row amount $size_{datarow}$ for training data $X_{train}$, the learning rate of the optimizer $l_r$.
for $i = 1 : epoch$
  $X = shuffle(X_{train})$
  For $j = 1 : cnt_{batch}$
    $X_{batch} = X[(j-1) \times size_{batch}, min(j \times size_{batch}, size_{datarow})]$
    Train data with Adam optimizer with $X_{batch}$
  Evaluate the network with $X_{train}$ and $X_{val}$
```

The shuffle operation is important, which can eliminate influence of the ordered data and improve the ability of generalization. In the experiment, *epoch* is firstly set to 1000, and the performance of the network would be observed. According to the performance on the validation data, *epoch* is set to the value which can make the validation data close to the best evaluation. And the values of other parameters such as $Size_{batch}$ and l_r are discussed in the experiment.

All the network in the experiment is training by TensorFlow, and the parameters, which are not mentioned, are set to the default value in TensorFlow.

4 Experiment and results

This section shows the experiment that FEIA framework with neural network solves CNOP of double gyre in ROMS. The problem and method can be reviewed in Sects. 2 and 3. The five feature dimensions, separately 20, 40, 60, 80 and 100, are tested in the experiment. As the reference data, the results that solved without reduction and solved with PCA are shown in Tables 1 and 2.

From the above table, it can be seen that the result without reduction is quite small and the corresponding solution has no probability to lead to double-gyre variation, which can be verified in the next experiment. With the reduction of PCA, the results show a significant improvement, and the effective solution can be obtained when the feature dimension is set to 40. However, it can also be seen that the probability obtaining the effective solution is low. This paper suggests two ways of applying neural network in FEIA framework, and the corresponding experiments and results are shown below.

4.1 The experiment for the first way

As shown in Sect. 3.2, the first way is training a network, which can serve as both the mapper and the re-constructor. AE and its three variants are tested, and the corresponding structures are shown in Table 3. One point needs to note is that the numbers of units in many layers are set to 256, which is limited by the machine's memory.

In the training stage, the batch size and the learning rate of the optimizer are adjusted to investigate their influence. Figures 6 and 7 show the related error (Eq. 26) varying curves of the validation data for the four networks within 1000 epochs.

$$related\_error = \frac{\|\mathbf{X} - r(p(\mathbf{X}))\|_F}{\|\mathbf{X}\|_F + e} \tag{26}$$

where $\boldsymbol{X}$ is the original data, $p(\cdot)$ is the mapper of the FEIA framework, $r(\cdot)$ is the re-constructor of the FEIA framework, $\|\cdot\|_F$ is the Frobenius norm and e is a small constant. The related error represents the difference rate between the original data and the reconstruct data, and it can be used to evaluate the quality of the network. It is worth noting that the related error is actually caused by the mapper rather than the re-constructor. Therefore, for FEIA framework, the only one related error appears in the stage constructing the initial feature solution.

From Fig. 7, it can be found that VAE cannot converge and has a large related error during 1000 epochs. This is because the feature generating process of VAE relies on the random, which leads to the instability. And it leads to that selecting the epoch with the proper related error becomes difficult, so VAE is not tested and discussed in the next experiments. From Figs. 6, 8 and 9, it can be found that (1) the batch size seems not to have a significant and same influence for the different networks, for example, with the increase in the batch size, the overfit degree decreases in the AE and SAE but increases in CAE; (2) the decrease in the learning rate can decrease the degree of oscillation and overfit. Meanwhile, from Figs. 6, 8 and 9, the epoch which can make the validation data close to the best related error is selected, and the corresponding networks used in FEIA framework are trained with the early stop. Tables 4 and 5 show the related errors and the single experiment results for the trained network.

From Tables 4 and 5, it can be found SAE has the best performance in the above trained networks. One possible reason might be that the size of the training data is only 2000, and the simple structure of the network could decrease the degree of the overfit. SAE has a relatively simple structure compared with AE and CAE, so it shows the best performance. And from Tables 4 and 5, it can be found that the batch size and learning rate seem not to have a significant influence on the results. Therefore, in the next experiments, the batch size is set to 40 and learning rate is set to 10^{-4}, which can avoid the influence of the extreme value on the judgment. On the other hand, the results in Table 6 verify that the first way of applying neural network (SAE) in FEIA framework is effective and even can solve the better solution. In order to further verify it, the statistic results of SAE for running ten times are shown in Table 7.

Compared with the results of PCA in Table 2, the results of SAE in Table 7 further verify the effectiveness of the first way. Compared with PCA, SAE can not only solve the results in the interval (1.55–1.65) where the solutions have the probability to lead to the variation, but also solve the results in the interval (> 1.65) where the solutions almost certainly lead to the variation. And the mean value and the max value of the results in Table 7 both are bigger than those in Table 2. On the other hand, in Table 2, PCA can

Table 1 The single experiment result of FEIA framework (PCA) and no reduction

	PCA					No Reduction
	20	40	60	80	100	
Result ($\times\ 10^{13}$ $\mathrm{m^5s^2}$)	1.520	**1.630**	1.384	1.420	1.402	1.214

The bold value represents the solution that can lead to double-gyre variation, which is same in Tables 4, 5 and 9, 10

Table 2 The statistic results of FEIA framework (PCA) for ten times

Dimension	Statistic ($\times 10^{13}$ m^{5}s^2)			Frequency (effective)		
	Max	Ave	Std	< 1.55	1.55 ~ 1.65	> 1.65
20	1.594	1.503	0.061	7(0)	3(0)	0(0)
40	1.630	1.535	0.092	5(0)	5(2)	0(0)
60	1.574	1.468	0.070	8(0)	2(0)	0(0)
80	1.507	1.418	0.047	10(0)	0(0)	0(0)
100	1.532	1.451	0.040	10(0)	0(0)	0(0)

"effective" represents the amount that the solutions can lead to double variation

Table 3 The tested network structures of the first way

	Structure
AE	fc(54,776) – fc(256) – fc(256) – fc(D_f) – fc(256) – fc(256) – fc(54,776)
SAE	fc(54,776) – fc(D_f) – fc(54,776)
CAE	(55, 110, 4) – conv(28, 28, 32) – conv(7, 7, 64) – fc(256) – fc(0.4 * D_f) – (56, 109, 4) – conv(28, 28, 32) – conv(7, 7, 64) – fc(256) – fc(0.4 * D_f) – (D_f) (56, 110, 1) – conv(28, 28, 32) – conv(7, 7, 64) – fc(256) – fc(0.2 * D_f) – – fc(0.4 * D_f) – fc(256) – fc(3136) – deconv(28, 28, 32) – deconv(55, 110, 4) – fc(0.4 * D_f) – fc(256) – fc(3136) – deconv(28, 28, 32) – deconv(56, 109, 4) – fc(0.2 * D_f) – fc(256) – fc(3136) – deconv(28, 28, 32) – deconv(56, 110, 1)
VAE	fc(54,776) – fc(256) – fc(256) – 2 × fc(D_f) – (D_f) – fc(256) – fc(256) – fc(54,776)

fc() represents the fully connected layer, conv() represents the convolutional layer, deconv() represents the deconvolutional layer, number in the parentheses represents the dimension of output and D_f represents the feature dimension, which are same in Table 8

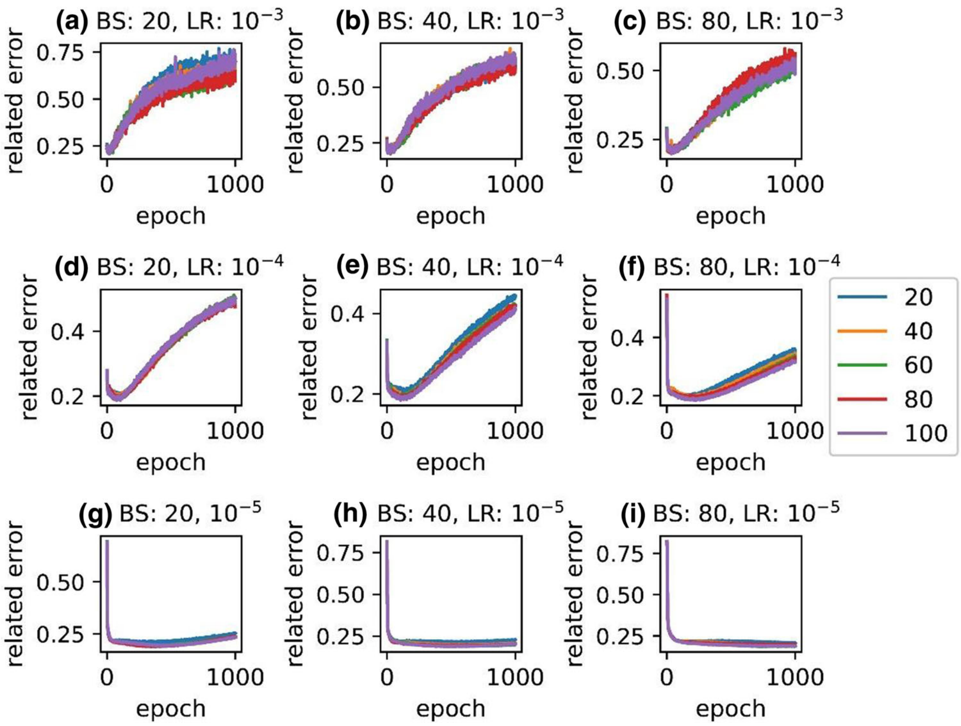


Fig. 6 The related error varying curves of validation data for AE (BS is the batch size and LR is the learning rate, which are same in Figs. 7, 8 and 9 and in Tables 4 and 5)

only help to obtain the effective solution when the feature dimension is 40, but SAE can help to obtain the effective solution except the feature dimension is 20. One possible reason is that the training of PCA is not for a special

Fig. 7 The related error varying curves of validation data for SAE

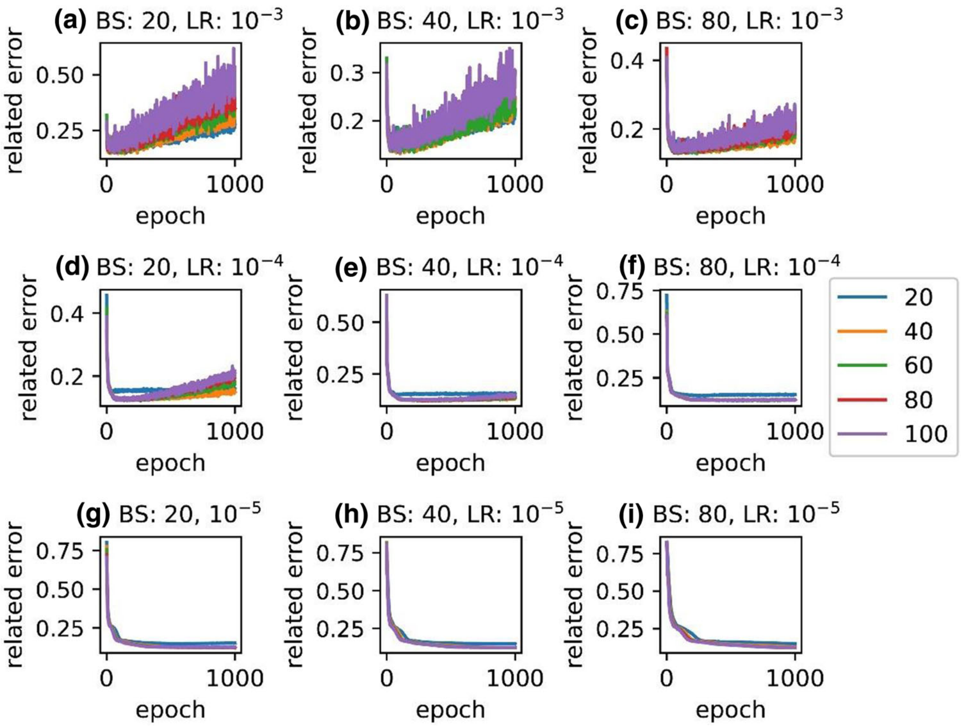
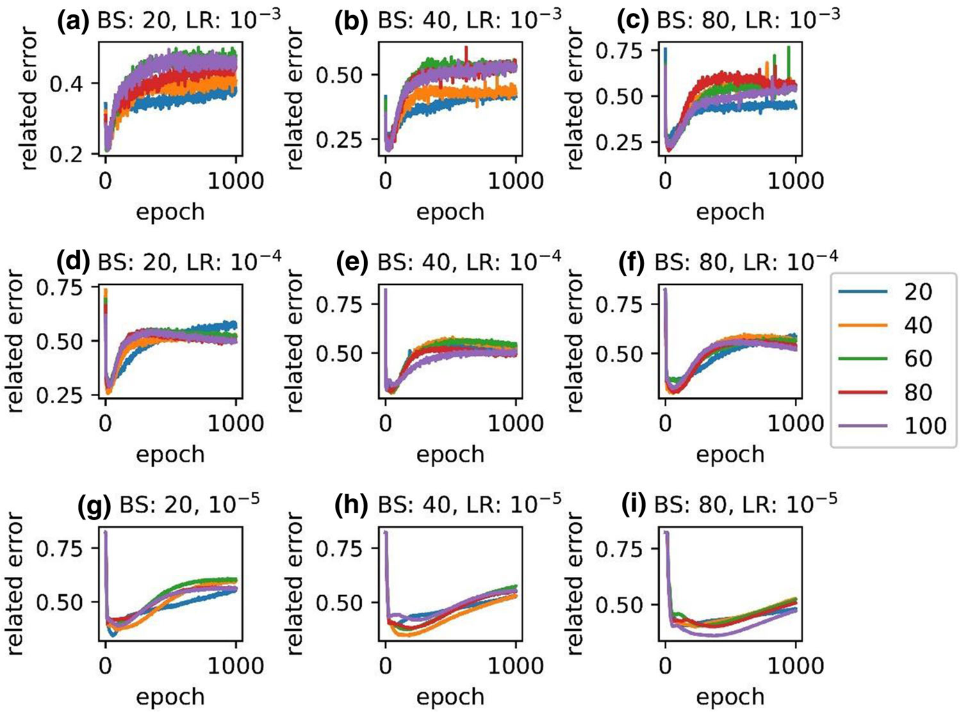


Fig. 8 The related error varying curves of validation data for CAE



feature dimension but the training of neural networks is for the special feature dimension. Therefore, compared with PCA, neural network might save the cost to determine the feature dimension.

4.2 The experiment for the second way

As shown in Sect. 3.3, the second way is combining PCA and the network, where PCA serves as the mapper and the network serves as the re-constructor. In this section,

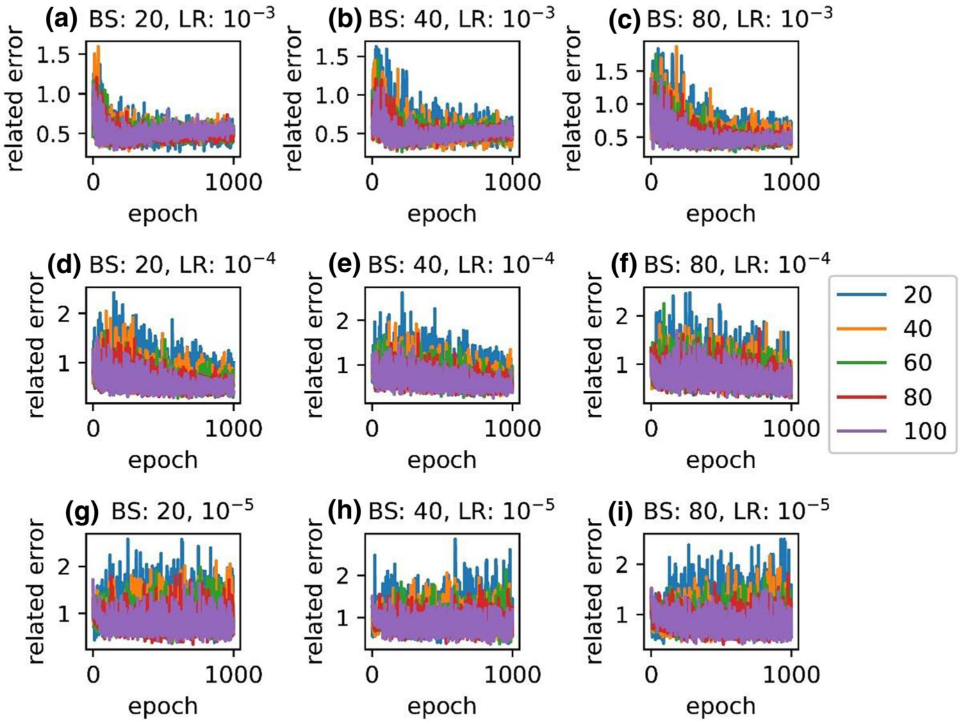


Fig. 9 The related error varying curves of validation data for VAE

Table 4 The related errors and the single experiment results for AE

Parameter		Related error					Result (× 10^{13} m^5s^2)				
BS	LR	20	40	60	80	100	20	40	60	80	100
20	10^{-3}	0.224	0.222	0.222	0.227	0.239	1.109	1.180	1.152	1.319	1.094
	10^{-4}	0.211	0.202	0.197	0.195	0.204	1.131	1.206	1.355	1.375	1.284
	10^{-5}	0.214	0.193	0.188	0.188	0.190	1.149	1.364	1.280	1.217	1.275
40	10^{-3}	0.207	0.220	0.215	0.228	0.226	1.195	1.272	1.164	1.098	1.129
	10^{-4}	0.214	0.188	0.202	0.190	0.193	1.143	1.200	1.285	1.251	1.289
	10^{-5}	0.209	0.196	0.197	0.192	0.193	1.125	1.231	1.297	1.305	1.267
80	10^{-3}	0.205	0.211	0.207	0.216	0.210	1.125	1.175	1.142	1.193	1.281
	10^{-4}	0.202	0.188	0.193	0.194	0.194	1.136	1.288	1.317	1.290	1.206
	10^{-5}	0.207	0.195	0.189	0.197	0.187	1.208	1.366	1.390	1.265	1.250

Table 5 The related errors and the single experiment results for SAE

Parameter		Related error					Result (× 10^{13} m^5s^2)				
BS	LR	20	40	60	80	100	20	40	60	80	100
20	10^{-3}	0.168	0.169	0.155	0.196	0.177	1.451	1.490	1.538	1.428	1.440
	10^{-4}	0.157	0.127	0.127	0.125	0.130	1.298	1.441	1.579	1.415	1.421
	10^{-5}	0.152	0.124	0.123	0.122	0.126	1.411	1.509	1.496	1.499	**1.624**
40	10^{-3}	0.159	0.150	0.142	0.138	0.152	1.475	1.435	1.324	**1.720**	1.541
	10^{-4}	0.157	0.123	0.124	0.124	0.124	1.473	1.566	1.461	1.358	**1.746**
	10^{-5}	0.148	0.123	0.123	0.123	0.124	1.437	1.538	1.553	1.442	1.404
80	10^{-3}	0.158	0.137	0.138	0.137	0.147	1.518	1.546	1.426	**1.620**	1.526
	10^{-4}	0.153	0.124	0.124	0.122	0.122	1.445	1.516	1.480	1.451	1.581
	10^{-5}	0.151	0.129	0.124	0.125	0.125	1.394	**1.647**	1.536	1.493	1.564

Table 6 The related errors and the single experiment results for CAE

Parameter		Related error					Result (× 10^{13} m^5s^2)				
BS	LR	20	40	60	80	100	20	40	60	80	100
20	10^{-3}	0.255	0.269	0.250	0.239	0.251	1.190	1.192	1.199	1.261	1.166
	10^{-4}	0.350	0.298	0.315	0.325	0.293	1.095	1.199	1.275	1.419	1.195
	10^{-5}	0.429	0.386	0.375	0.401	0.398	1.107	1.152	1.253	1.162	1.223
40	10^{-3}	0.267	0.220	0.223	0.232	0.211	1.139	1.325	1.254	1.253	1.167
	10^{-4}	0.346	0.331	0.328	0.326	0.319	1.156	1.144	1.415	1.116	1.242
	10^{-5}	0.389	0.394	0.380	0.421	0.391	9.771	1.128	1.358	1.262	1.119
80	10^{-3}	0.255	0.215	0.218	0.227	0.220	1.134	1.374	1.232	1.311	1.242
	10^{-4}	0.352	0.334	0.346	0.319	0.307	1.125	1.243	1.140	1.223	1.286
	10^{-5}	0.432	0.385	0.376	0.380	0.397	1.121	1.180	1.352	1.241	1.263

Table 7 The statistic results of FEIA framework (SAE) for ten times

Dimension	Statistic (× 10^{13} m^5s^2)			Frequency (effective)		
	Max	Ave	Std	< 1.55	1.55 ~ 1.65	> 1.65
20	1.502	1.455	0.040	10(0)	0(0)	0(0)
40	1.681	1.576	0.082	2(0)	6(2)	2(2)
60	1.710	1.527	0.109	6(0)	2(1)	2(2)
80	1.662	1.514	0.093	7(0)	2(0)	1(1)
100	1.746	1.519	0.123	6(0)	3(0)	1(1)

decoder and GAN are tested to serve as the re-constructor, and the corresponding structures are shown in Table 8.

Because the simple structure obtains the better results in Sect. 4.1, the number of the extra latent layer is adjusted to investigate its influence. Figures 10 and 11 show the related error varying curves of the validation data for the two networks within 1000 epochs. And with the best epochs selected by Figs. 10 and 11, the related errors and the single experiment results for the trained network are shown in Tables 9 and 10.

From Figs. 10 and 11, it can be found that decreasing the number of the extra latent layers can decrease the degree of oscillation and overfit. And from Tables 9 and 10, it can be found that decreasing the number of extra latent layers can obtain better related errors and objective function values. These results further verify the conclusion in Sect. 4.1, if the size of the training sample set is small, the simpler structure would be proper for FEIA framework. On the other hand, the results of solving CNOP are similar for the two structures. And the two structures can both obtain the effective solutions leading to double-gyre variation, which verifies the effectiveness of the second way. Similar to Sect. 4.1, GAN (EL = 0) are selected to test ten times to further verify the performance of the second way, and the results are shown in Table 11.

The results in Table 11 are similar to those in Table 7. Compared with PCA, the results with the second way show the following chrematistics: (1) The solution in better interval can be obtained; (2) better average value and max value of the objective function can be obtained; and (3) the effective solution can be obtained in more feature dimension. These results further show the effectiveness of the second way.

4.3 The discussion for the result

The above experiments verify the effectiveness of neural network in FEIA framework. Both the first way and the second way can solve the effective solution which can lead to double-gyre variation. And with the good training, FEIA framework with neural network can even show a better performance than FEIA framework with PCA. According to the above experiments, the optimization details of the network and the performance of FEIA framework neural network are summarized and discussed.

4.3.1 Structure and parameter selection

In the experiments, the influences of the batch size, the learning rate and the complex degree of the network are tested. With the condition that the size of the training data is relatively small, which is 2000 × 54,776, the influence of them can be summarized as follows: (1) The complexity degree of the network has the largest influence, while the batch size is the least sensitive, and the influence of the learning rate is between them. (2) One of the main reasons

Table 8 The tested network structures of second way

	Structure
Decoder	fc(D_f) – EL × (fc(256) –) fc(54,776)
GAN	Generator:
	fc(D_f) – EL × (fc(256) –) fc(54,776)
	Discriminator:
	fc(54,776) – fc(256) – fc(256) – fc(1)

EL is the number of the extra latent layer, which is same in Figs. 10–11

to cause the performance loss is the overfit, and the decrease in the complexity degree and learning rate can decrease the degree of oscillation and overfit. (3) Although the learning rate can also decrease the degree of overfit, it does not obtain a smaller error in the best epoch. Therefore, with the early stop, only the change in the complexity degree can cause the relatively significant optimization. Above all, in order to obtain the proper mapper and re-constructor for FEIA framework, the structure of the network needs to be considered firstly according to the characteristic and size of the data. And then, the adjusting of training process, such as decreasing the learning rate, might give a further optimization.

4.3.2 Performance analysis

In fact, the process of PCA is generally faster than that of training a network. However, in some problem, such as solving CNOP of double gyre in the experiment, the main time cost is on the calculation of intelligent algorithm. In this paper, because the calculation of the objective function involves in the integration of the model, even the time cost of CAE, which is the highest for the network trained in the experiments, is lower than that of intelligent algorithm. Therefore, training the proper mapper and re-constructor is relatively more important than costing a less time to obtain the mapper and constructor. And in the experiments, neural network shows three advantages compared with PCA: (1) Neural network can help to solve the solutions in better interval where the solutions almost certainly lead to the variation. (2) The solutions solved with FEIA framework with neural network have a larger mean value and max value. (3) The effective solutions, which can lead to the variation, can be solved in more feature dimension. The first two points suggest that neural network with proper design can construct a better mapping-reconstruction structure to help FEIA framework solve the problem. And the last point shows that neural network might save the cost to determine the feature dimension, which means the calculation frequency of intelligent algorithm part might be decreased. These might be because neural network can do a more special fitting compared with PCA. For example, PCA is not training for a special feature dimension, but neural network is training for the special feature dimension. Above all, according to the results of the experiments, neural network is proven to be an effective component which can be applied in FEIA framework. And with the proper design, the performance of FEIA framework with neural network might be better than that of FEIA framework with the classical method.

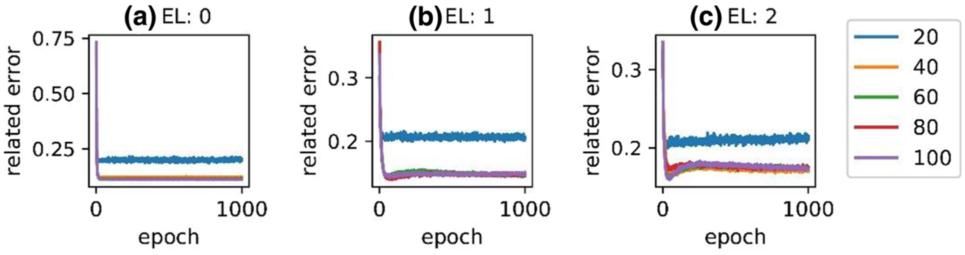


Fig. 10 The related error varying curves of validation data for decoder

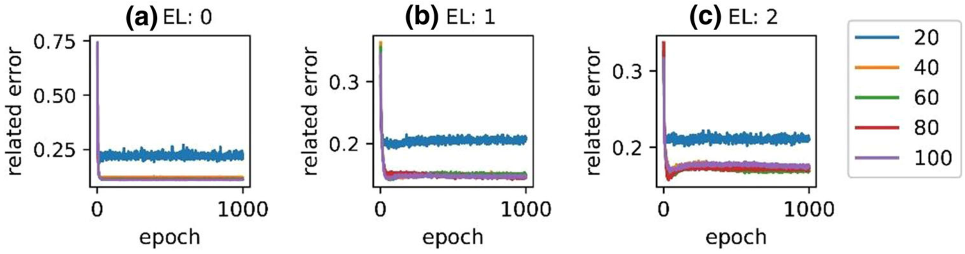


Fig. 11 The related error varying curves of validation data for GAN

Table 9 The related errors and the single experiment results for decoder

Parameter	Related error					Result ($\times 10^{13}$ m^{5}s^2)				
EL	20	40	60	80	100	20	40	60	80	100
0	0.197	0.122	0.115	0.111	0.112	1.529	1.409	1.476	1.413	1.466
1	0.204	0.149	0.146	0.150	0.147	1.358	**1.605**	1.425	1.360	1.384
2	0.209	0.159	0.163	0.167	0.175	1.318	1.241	1.329	1.237	1.265

Table 10 The related errors and the single experiment results for GAN

Parameter	Related error					Result ($\times 10^{13}$ m^{5}s^2)				
EL	20	40	60	80	100	20	40	60	80	100
0	0.222	0.122	0.116	0.114	0.112	1.489	**1.688**	1.499	1.501	**1.640**
1	0.205	0.147	0.147	0.141	0.147	1.239	1.553	1.396	1.334	1.469
2	0.210	0.170	0.172	0.164	0.172	1.201	1.296	1.429	1.369	1.220

Table 11 The statistic results of FEIA framework (GAN) for ten times

Dimension	Statistic ($\times 10^{13}$ m^{5}s^2)			Frequency (effective)		
	Max	Ave	Std	< 1.55	$1.55 \sim 1.65$	> 1.65
20	1.491	1.440	0.043	10(0)	0(0)	0(0)
40	1.820	1.566	0.135	6(0)	1(0)	3(3)
60	1.683	1.541	0.076	6(0)	2(0)	2(2)
80	1.657	1.495	0.071	8(0)	1(0)	1(1)
100	1.640	1.530	0.096	5(0)	5(2)	0(0)

5 Conclusion

In this paper, the two ways of applying neural work in FEIA framework are suggested. The first way is training a network to serve as both mapper and re-constructor, and the second way is using the classical method to serve as the mapper and training a network to serve as the corresponding re-constructor of the mapper. With the experiments solving CNOP of double-gyre variation in ROMS, how to train the proper neural network in FEIA is discussed, and the good performance of FEIA framework with neural network is verified. Compared with PCA, neural network can construct a better mapping-reconstruction structure with the proper design. Therefore, the solutions solved by FEIA framework with neural network obtain better objective values and have a larger probability leading to the expected physical phenomenon.

In fact, besides CNOP problem, FEIA framework can be applying in many other problems. And in this paper, the size of training data for the network is relatively small. It is worth to look into the performance of FEIA framework in more problems and the way applying the network with more data.

Acknowledgements The authors would like to thank other members in the research group for helpful and valuable suggestions.

Funding This study is supported in part by the National Key Research and Development Program of China under Grant 2020YFA0608002, in part by the National Natural Science Foundation of China under Grant 42075141, in part by the Key Project Fund of Shanghai 2020 "Science and Technology Innovation Action Plan" for Social Development under Grant 20dz1200702, and in part by the Fundamental Research Funds for the Central Universities under Grant 13502150039/003.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent Informed consent was obtained from all individual participants included in the study.

References

Abdi H, Williams LJ (2010) Principal component analysis. WIREs. Comput Stat 2:433–459. https://doi.org/10.1002/wics.101

Chen Z, Yeo CK, Lee BS, Lau CT (2018) Autoencoder-based network anomaly detection. In: 2018 Wireless telecommunications symposium (WTS), pp. 1–5

Creswell A, White T, Dumoulin V et al (2018) Generative adversarial networks: an overview. IEEE Signal Process Mag 35:53–65. https://doi.org/10.1109/MSP.2017.2765202

Goodfellow ., Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014). Generative adversarial nets. In: Advances in neural information processing systems

Hinton GE (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507. https://doi.org/10.1126/science.1127647

Jiang L, Duan W (2020) Target observation of mesoscale eddies in the ocean. InL EGU general assembly conference abstracts (p. 6633). https://doi.org/10.5194/egusphere-egu2020-6633

Jiao Z, Ji Y, Gao P, Wang S-H (2020) Extraction and analysis of brain functional statuses for early mild cognitive impairment using variational auto-encoder. J Amb Intell Human Comput. https://doi.org/10.1007/s12652-020-02031-w

Kingma DP, Welling M (2014) Auto-encoding variational Bayes. In: 2nd International conference on learning representations, ICLR 2014—conference track proceedings

Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. In: 3rd International conference on learning representations, ICLR 2015—conference track proceedings. Retrieved from http://arxiv.org/abs/1412.6980

Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continuous engineering optimization harmony search theory and practice. Comput Methods Appl Mech Eng 194(36):3902–3933. https://doi.org/10.1016/j.cma.2004.09.007

Lin E, Mukherjee S, Kannan S (2020) A deep adversarial variational autoencoder model for dimensionality reduction in single-cell RNA sequencing analysis. BMC Bioinforms 21:64. https://doi.org/10.1186/s12859-020-3401-5

Liu J, Wang S, Yang W (2019) Sparse autoencoder for social image understanding. Neurocomputing 369:122–133. https://doi.org/10.1016/j.neucom.2019.08.083

Liu G, Xie L, Chen C-H (2020a) Unsupervised text feature learning via deep variational auto-encoder. ITC 49:421–437. https://doi.org/10.5755/j01.itc.49.3.25918

Liu J, Guo W, Cui B, et al (2020) Targeted observations based on identified sensitive areas by CNOP to improve the thermal structures prediction in the summer Yellow Sea: preparatory work for the campaign in the field. In EGU General Assembly Conference Abstracts (p. 12376). https://doi.org/10.5194/egusphere-egu2020-12376

Masci J, Meier U, Cireşan D, Schmidhuber J (2011) Stacked convolutional auto-encoders for hierarchical feature extraction. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and s) https://doi.org/10.1007/978-3-642-21735-7_7

Moore AM, Arango HG, Di Lorenzo E, Cornuelle BD, Miller AJ, Neilson DJ (2004) A comprehensive ocean prediction and analysis system based on the tangent linear and adjoint of a regional ocean model. Ocean Model 7(1–2):227–258. https://doi.org/10.1016/j.ocemod.2003.11.001

Mu M, Duan WS, Wang B (2003) Conditional nonlinear optimal perturbation and its applications. Nonlinear Process Geophys 10(6):493–501. https://doi.org/10.5194/npg-10-493-2003

Mu B, Ren J, Yuan S, Zhou F (2019) Identifying typhoon targeted observations sensitive areas using the gradient definition based method. Asia-Pac J Atmos Sci 55(2):195–207. https://doi.org/10.1007/s13143-018-0068-1

Mu B, Wen S, Yuan S, Li H (2015) PPSO: PCA based particle swarm optimization for solving conditional nonlinear optimal perturbation. Comput Geosci 83:65–71. https://doi.org/10.1016/j.cageo.2015.06.016

Mu M, Zheng Q (2005) Zigzag oscillations in variational data assimilation with physical "On–Off" processes. Mon Weather Rev 133(9):2711–2720. https://doi.org/10.1175/MWR2995.1

Nauw JJ, Dijkstra HA (2001) The origin of low-frequency variability of double-gyre wind-driven flows. J Mar Res 59(4):567–597. https://doi.org/10.1357/002224001762842190

Ng A (2011) Sparse autoencoder. CS294A Lecture Notes, 72(2011):1–19

Qiu B (2000) Interannual variability of the Kuroshio extension system and its impact on the wintertime SST field. J Phys Oceanogr 30(6):1486–1502. https://doi.org/10.1175/1520-0485(2000)030%3c1486:IVOTKE%3e2.0.CO;2

Ramamurthy M, Robinson YH, Vimal S, Suresh A (2020) Auto encoder based dimensionality reduction and classification using convolutional neural networks for hyperspectral images. Microprocess Microsyst 79:103280. https://doi.org/10.1016/j.micpro.2020.103280

Ringnér M (2008) What is principal component analysis? Nat Biotechnol 26:303–304. https://doi.org/10.1038/nbt0308-303

Rumelhart DE, Hinton GE, Williams RJ (1986) Learning internal representations by error propagation. In: Rumelhart DE, Mcclelland JL (eds) Parallel distributed processing: explorations in the microstructure of cognition, vol 1. Foundations. MIT Press, Cambridge, MA, pp 318–362

Schonfeld E, Schiele B, Khoreva A (2020) A U-Net based discriminator for generative adversarial networks. In: 2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR). IEEE, Seattle, WA, USA, pp 8204–8213

Shchepetkin AF, McWilliams JC (2005) The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model. Ocean Model 9(4):347–404. https://doi.org/10.1016/j.ocemod.2004.08.002

Shen J, Medjo TT, Wang S (1999) On a wind-driven, double-gyre, quasi-geostrophic ocean model: numerical simulations and structural analysis. J Comput Phys 155(2):387–409. https://doi.org/10.1006/jcph.1999.6344

Sun G, Mu M, Zhang Y (2010) Algorithm studies on how to obtain a conditional nonlinear optimal perturbation (CNOP). Adv Atmos Sci 27(6):1311–1321. https://doi.org/10.1007/s00376-010-9088-1

Towara M, Naumann U (2013) A discrete adjoint model for OpenFOAM. Proced Comput Sci 18:429–438. https://doi.org/10.1016/j.procs.2013.05.206

Wang Q, Mu M, Sun G (2020) A useful approach to sensitivity and predictability studies in geophysical fluid dynamics: conditional non-linear optimal perturbation. Natl Sci Rev 7:214–223. https://doi.org/10.1093/nsr/nwz039

Wang B, Tan X (2009) A fast algorithm for solving CNOP and associated target observation tests. Acta Meteor Sin 23(4):387–402

Wang Y, Yao H, Zhao S (2016) Auto-encoder based dimensionality reduction. Neurocomputing 184:232–242. https://doi.org/10.1016/j.neucom.2015.08.104

Wold S, Esbensen K, Geladi P (1987) Principal component analysis. Chemom Intell Lab Syst 2(1–3):37–52. https://doi.org/10.1016/0169-7439(87)80084-9

Xie L, Liu G, Lian H (2019) Deep variational auto-encoder for text classification. In: 2019 IEEE International conference on industrial cyber physical systems (ICPS). pp 737–742

Yuan S, Li M, Wang Q, Zhang K, Zhang H, Mu B (2019a) Optimal precursors of double-gyre regime transitions with an adjoint-free method. J Oceanol Limnol 37(4):1137–1153. https://doi.org/10.1007/s00343-019-7235-9

Yuan S, Zhang H, Li M, Mu B (2019b) CNOP-P-based parameter sensitivity for double-gyre variation in ROMS with simulated annealing algorithm. J Oceanol Limnol. https://doi.org/10.1007/s00343-019-7266-2

Zhang K, Mu M, Wang Q (2015) The impact of initial error on predictability of Double-gyre variability. Mar Sci 39(5):120–128. https://doi.org/10.11759/hykx20130304001

Zhang K, Mu M, Wang Q, Yin B, Liu S (2019c) CNOP-Based adaptive observation network designed for improving upstream Kuroshio transport prediction. J Geophys Res: Oceans 124(6):4350–4364. https://doi.org/10.1029/2018JC014490

Zhang RH, Tao LJ, Gao C (2018) An improved simulation of the 2015 El Niño event by optimally correcting the initial conditions and model parameters in an intermediate coupled model. Clim Dyn. https://doi.org/10.1007/s00382-017-3919-z

Zhang H, Xu T, Li H et al (2019b) StackGAN++: realistic image synthesis with stacked generative adversarial networks. IEEE Trans Pattern Anal Mach Intell 41:1947–1962. https://doi.org/10.1109/TPAMI.2018.2856256

Zhang K, Zhang J, Ma X et al (2021) History matching of naturally fractured reservoirs using a deep sparse autoencoder. SPE J 26:1700–1721. https://doi.org/10.2118/205340-PA

Zhang H, Goodfellow I, Metaxas D, Odena A (2019a) Self-Attention Generative Adversarial Networks. In: Proceedings of the 36th international conference on machine learning. PMLR, pp 7354–7363

Zheng Q, Sha J, Shu H, Lu X (2014) A variant constrained genetic algorithm for solving conditional nonlinear optimal perturbations. Adv Atmos Sci 31(1):219–229. https://doi.org/10.1007/s00376-013-2253-6

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.