Advances in Applied Mathematics 应用数学进展, 2022, 11(5), 3134-3149
Published Online May 2022 in Hans. http://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2022.115333

Hans汉斯

基于广义化自适应的NAG方法非负张量分解模型

陶名康

上海理工大学，上海

收稿日期：2022年4月27日；录用日期：2022年5月21日；发布日期：2022年5月31日

摘　要

单元素非负乘法更新算法在学习模型超参数时会出现长尾收敛的情况，本文通过将NAG方法融入到单元素非负乘法更新算法中，得到了广义化的NAG方法，并在此基础上提出了基于广义化自适应的NAG非负张量分解模型。在训练过程中利用粒子群算法对模型的正则化系数和算法的加速度系数进行了优化。最后，在两个真实的工业数据集上的对比实验表明，本文提出的广义化NAG方法明显提高了模型的收敛速度。

关键词

非负乘法更新，NAG方法，高维稀疏数据，张量分解

Non-Negative Tensor Decomposition Model Based on Generalized and Adaptive NAG Method

Mingkang Tao

University of Shanghai for Science & Technology, Shanghai

Received: Apr. 27th, 2022; accepted: May 21st, 2022; published: May 31st, 2022

Abstract

In order to solve the problem of long tail convergence when SLF-NMU algorithm learns model hyperparameters, the NAG method is integrated into the SLF-NMU algorithm, and the generalized NAG method is obtained. On this basis, a non-negative tensor decomposition based on generalized

文章引用: 陶名康. 基于广义化自适应的 NAG 方法非负张量分解模型[J]. 应用数学进展, 2022, 11(5): 3134-3149.
DOI: 10.12677/aam.2022.115333

and adaptive NAG method is proposed. During the training process, particle swarm optimization is used to optimize parameters. Comparison with three similar algorithms on real industrial data shows that the proposed model can improve the convergence speed.

Keywords

Non-Negative and Multiplicative Update, NAG Method, High Dimensional and Sparse Data, Tensor Decomposition

1. 引言

随着云计算的快速发展，许多软件供应商提供了大量具有类似功能的 Web 服务，从这些具有类似功能的 Web 服务中为用户选择最合适的服务成为了一个热门问题[1] [2]。Web 服务可以通过用户服务质量(quality-of-service, QoS)来评估。其中，基于用户的 QoS 数据为用户挑选适合的 Web 服务提供了重要的依据。

各种各样的方法[3] [4] [5]被提出来解决这个问题，Koren [6]将动态数据沿着时间轴分割成不相交的时间切片，其中每一个切片表示用户在某个时间点对一些服务的调用和交互的数据，然后在每一个时间切片上建立潜在因子分析模型。该模型相比于没有考虑时间的潜在因子分析模型的性能要好，但是该模型反应的时间信息被大大限制，因为如果沿着时间轴将数据切片，将切断每一个时间点之间的内在联系。因此在为动态数据建立模型的过程中应该充分考虑这些时间点之间的关系。

张量分解(Tensor Decomposition, TD)模型[7]将不再遭受这个问题，TD 模型在高维稀疏(High Dimensionaland Sparse, HiDS)张量上模拟用户－服务－时间(User-Service-Time)之间的交互，然后再进行潜在因子分解。虽然，TD 模型将时间信息建模成不同的潜在因子(LatentFactor, LFs)，但是它的训练过程是整个动态数据集，而不是时间切片。TD 模型考虑了各个时间之间的联系，从而可以更好地刻画隐藏在 HiDS 张量中的信息。

以前的研究[8] [9] [10] [11]表明，Nesterov 加速梯度下降法(Nesterov's Accelerated Gradient, NAG)在加速基于梯度下降法的学习算法的收敛速度非常有效。与动量法相比 NAG 方法更新过程中更细微的融合历史梯度信息。此外，该方法减弱了训练过程中的震荡，因此可以保证模型更快速的收敛。

基于以上发现，本文提出了基于广义化自适应的 NAG 非负张量分解模型(NHLFT-NAG)。该模型采用张量来代表随时间变化的 QoS 数据，因此把数据随时间变化的信息考虑到模型中。

本文的创新点如下：

1) 将广义化的 NAG 方法融入到 SLF-NMU [7]算法中，提出了广义化的 NAG 算法来提高学习潜在因子的效率；2) 对用户，服务，时间分别建立了线性偏置，来处理 QoS 数据的波动性，从而使模型具有更好地鲁棒性和更高的预测精度；3) 通过在训练过程中加入粒子群算法(PSO)，使得模型正则化系数和算法的加速系数可以自适应调节。

2. 相关工作

在分析和预测动态的 QoS 数据时，常将用户－服务－时间的张量作为基本的输入数据源。QoS 数据

定义在非负实数域上并含有缺失值如图 1 所示。本文首先定义目标张量。

定义 1 (HiDS 用户 - 服务 - 时间张量)：给出 I，J 和 K，$\boldsymbol{Y}^{|I|\times|J|\times|K|}$ 是一个张量，张量中每一个元素 y_{ijk} 表示某一个用户 $i\in I$，在某个时间 $k\in K$ 点击某一个服务 $j\in J$。其中 $\boldsymbol{Y}$ 中已知和未知元素集合用 Λ 和 Γ 来表示，当 $|\Lambda|\ll|\Gamma|$ 时 $\boldsymbol{Y}$ 是 HiDS。

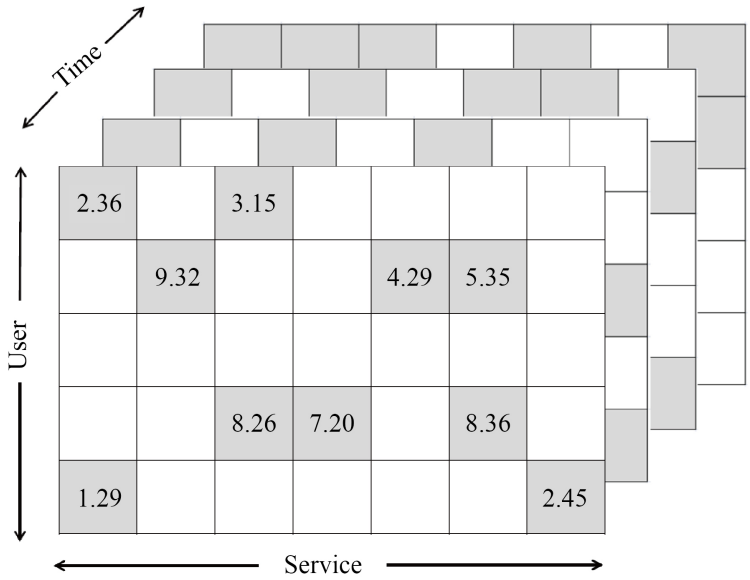


Figure 1. Tensor of user-service-time (blank part indicates missing value)
图 1. 用户 - 服务 - 时间张量(空白部分表示缺失)

2.1. 高阶的潜在因子模型

本文采用的是 Charu C [12]提出了一种特殊的张量分解结构，评分张量 $\boldsymbol{Y}^{|I|\times|J|\times|K|}$ 可以被分解为三个矩阵分别是：用户因子矩阵 $U^{|I|\times f}$，项因子矩阵 $V^{|J|\times f}$，环境因子矩阵 $W^{|K|\times f}$，其中 $f\ll\min\{|I|,|J|,|K|\}$ 称为潜在因子维数。这种张量分解结构其实是基于非负矩阵分解的潜在因子模型的一种高阶表现形式。此时评分张量 $\boldsymbol{Y}$ 任意位置上元素的估计值可以表示为：

$$\hat{y}_{ijk}=\left(UV^{\mathrm{T}}\right)_{ij}+\left(UW^{\mathrm{T}}\right)_{ik}+\left(VW^{\mathrm{T}}\right)_{jk}=\sum_{r=1}^{f}\left(u_{ir}v_{jr}+u_{ir}w_{kr}+v_{jr}w_{kr}\right) \tag{1}$$

其中 u_{ir},v_{jr},w_{kr} 表示 U,V,W 中的元素。

为了得到上式中的 U，V 和 W，在已知集合 Λ 上利用欧式距离建立损失函数来衡量 $\boldsymbol{Y}$ 和 $\hat{\boldsymbol{Y}}$ 的差异：

$$\arg\min_{U,V,W} Loss=\sum_{y_{ijk}\in\Lambda}\left(y_{ijk}-\hat{y}_{ijk}\right)^2 \tag{2}$$

为了防止训练过程中出现过拟合在(2)中加入正则化项可以提高模型的性能，得到(3)式：

$$\arg\min_{U,V,W} Loss=\sum_{y_{ijk}\in\Lambda}\left(\left(y_{ijk}-\hat{y}_{ijk}\right)^2+\lambda\sum_{r=1}^{f}\left(u_{ir}^2+v_{jr}^2+w_{kr}^2\right)\right) \tag{3}$$

其中 λ 表示正则化系数。

由于 QoS 数据通常具有非负性，因此对 U，V 和 W 做非负限制，得到了最终的损失函数如下所示：

$$\begin{aligned}&\arg\min_{U,V,W} Loss = \sum_{y_{ijk}\in\Lambda}\left(\left(y_{ijk}-\hat{y}_{ijk}\right)^2+\lambda\sum_{r=1}^{f}\left(u_{ir}^2+v_{jr}^2+w_{kr}^2\right)\right)\\&\text{s.t. } \forall i\in I,\ j\in J,\ k\in K,\ r\in\{1,2,\cdots,f\}\\&\quad u_{ir}\geq 0,\ v_{jr}\geq 0,\ w_{kr}\geq 0.\end{aligned}\tag{4}$$

2.2. NAG 方法

在目标函数的求解中，梯度下降法是一种常用的方法，它的缺点是在迭代的过程中容易陷入鞍点，或者陷入局部最优，导致收敛的速度缓慢。为了解决这个问题，Yurii Nesterov [8]提出了梯度下降法的一个改进算法，即 Nesterov 加速梯度下降法(Nesterov's Accelerated Gradient, NAG)，NAG 方法更新公式如下：

$$\begin{cases}u_0=0\\u_{(t)}=\mu u_{(t-1)}-\eta\nabla_\theta L\left(\theta_{(t-1)}+\mu u_{(t-1)}\right)\\\theta_{(t)}\leftarrow\theta_{(t-1)}+u_{(t)}\end{cases}\tag{5}$$

其中 $L(\theta)$ 表示自变量为 θ 损失函数，u 表示 θ 的更新速度，分别 u_0 表示 u 在训练过程中的初始，$u_{(t)},u_{(t-1)}$ 表示第 t 次和第$(t-1)$次的状态，$\theta_{(t)}$ 和 $\theta_{(t-1)}$ 表示训练过程中第 t 次和第$(t-1)$次的状态，μ 表示加速系数。

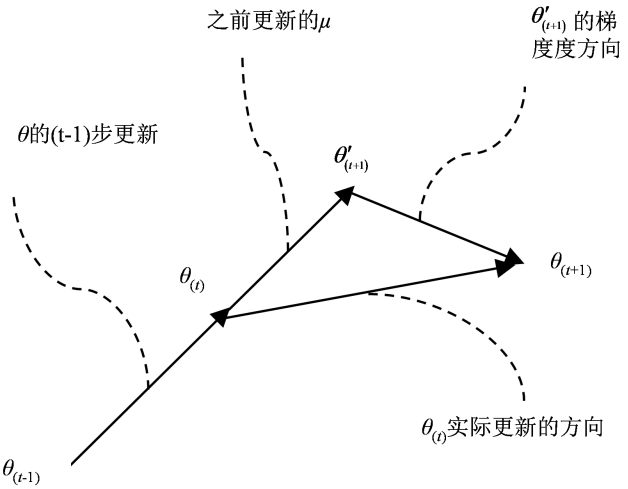


Figure 2. Diagram of NAG method
图 2. NAG 方法示意图

从(5)式和图 2 可以看到 NAG 方法包涵两个步骤：1) 沿着之前更新的方向移动 θ 得到一个中间状态 $\theta'_{(t+1)}$，计算中间状态的梯度，此时计算 $\theta'_{(t+1)}$ 的梯度时包含了更多历史梯度的信息；2) 通过线性组合这个梯度和之前的更新速度得到最终的更新方向。

2.3. 广义化的 NAG 方法

从(5)式可以看到 NAG 方法显式的采用了梯度，但是单元素非负乘法更新[7] (single latent factor-dependent, non-negative and multiplicative update, SLF-NMU)算法由于对学习率的设置，隐式的采用了梯度。那么如何把 NAG 方法融入到 SLF-NMU 学习方法中得到一个广义化的 NAG 方法呢？根据[8]令

$u_{(t-1)}=\theta_{(t-1)}-\theta_{(t-2)}$ 然后重新改写(5)式如下：

$$\begin{cases}\theta'_{(t-1)}=\begin{cases}\theta_{(0)};t=1\\ \theta_{(t-1)}+\mu\left(\theta_{(t-1)}-\theta_{(t-2)}\right);t\geq 2\end{cases}\\ \theta_{(t)}\leftarrow\theta'_{(t-1)}-\eta\nabla L_{\theta'}\left(\theta'_{(t-1)}\right)\end{cases}\tag{6}$$

其中 $\theta'_{(t-1)}$ 表示 θ 的中间状态包含了之前的更新速度和第$(t-1)$次训练迭代后的状态。

在(6)式中可以发现使用 NAG 方法更新参数时包涵以下参数：1) 原始状态的参数，如 $\theta_{(t-1)}$；2) 中间状态的参数，如 $\mu\left(\theta_{(t-1)}-\theta_{(t-2)}\right)$；3) 更新参数时使用的算法，其中(6)式中使用的算法是梯度下降法。

实际上，在隐式依赖梯度的算法中，可以更容易的获得更新增量。令 $\theta''_{(t)}$ 是 θ' 在采用某个算法第 t 次迭代后的状态即：

$$\theta''_{(t)}\xleftarrow{\text{Algorithm}}\arg\min_{w}L_{\theta'}\left(\theta'_{(t-1)}\right)\tag{7}$$

因此更新的增量 $\Delta_{(t)}$ 为:

$$\Delta_{(t)}=\theta''_{(t)}-\theta'_{(t-1)}\tag{8}$$

为了得到 $\theta_{(t)}$ 的广义形式，令 $\Delta_{(t)}=-\eta\nabla L_{\theta'}\left(\theta'_{(t-1)}\right)$，可以得到下式：

$$\theta_{(t)}=\theta'_{(t-1)}+\Delta_{(t)}=\theta'_{(t-1)}+\left(\theta''_{(t)}-\theta'_{(t-1)}\right)=\theta''_{(t)}\tag{9}$$

在梯度下降法中，(6)式和(9)式都等效地将 NAG 方法纳入到训练过程中。因此，(6)式与(9)式结合得到广义 NAG 方法：

$$\begin{cases}\theta'_{(t-1)}=\begin{cases}\theta_{(0)};t=1\\ \theta_{(t-1)}+\mu\left(\theta_{(t-1)}-\theta_{(t-2)}\right);t\geq 2\end{cases}\\ \theta_{(t)}\xleftarrow{\text{Algorithm}}\arg\min_{\theta}L\left(\theta'_{(t-1)}\right)\end{cases}\tag{10}$$

3. NHLFT-NAG 模型

3.1. 带偏置 NHLFT 模型

推荐系统中的数据会随着时间的波动，根据[13]，处理这种具有波动数据时，在潜在因子模型中加入线性偏置(Linear Bias, LB)，提高了模型的鲁棒性和预测的准确性。类似的在 NHLFT 中也加入线性偏置(以 QoS 张量数据为例)，分别对用户，服务和时间建立 LBs。

该模型的偏置和三个矩阵即用户矩阵 $D^{|I|\times 3}$，第一列元素记为 d_{i1}，服务矩阵 $E^{|J|\times 3}$，第二列元素记为 e_{j2}，时间矩阵 $F^{|K|\times 3}$，第三列元素记为 f_{k3}。每一个矩阵只有一列是 LBs 的值，其他两列均填充了常数 1 如图 3 所示。

提取三个矩阵中的第一列，可以得到三个列向量，然后把这三个列向量做外积(Outer Product)，得到张量 L_1。对用户 i，L_1 中的元素记为 l^1_{ijk}，由向量外积的计算公式可知 $\forall j\in J,k\in K:l^1_{ijk}=d_{i1}$，同理可以得到 L_2,L_3。令 $\forall i\in I:d_{i1}=a_i$，$\forall j\in J:e_{j2}=b_j$，$\forall k\in K:f_{k3}=c_k$，由 a_i,b_j,c_k 组成的向量分别记做 A,B,C，再根据(4)式得到带偏置的目标张量中估计值的表达式：

$$\hat{y}_{ijk}=\sum_{r=1}^{f}\left(u_{ir}v_{jr}+u_{ir}w_{kr}+v_{jr}w_{kr}\right)+a_i+b_j+c_k\tag{11}$$

最终的损失函数如下：

$$\begin{aligned}&\arg\min_{U,V,W} Loss = \sum_{y_{ijk}\in\Lambda}\left(\left(y_{ijk}-\hat{y}_{ijk}\right)^2+\lambda_1\sum_{r=1}^{f}\left(u_{ir}^2+v_{jr}^2+w_{kr}^2\right)+\lambda_2\left(a_i^2+b_j^2+c_k^2\right)\right)\\&\text{s.t. } \forall i\in I, j\in J, k\in K, r\in\{1,2,\cdots,f\}\\&\quad u_{ir}\geq 0, v_{jr}\geq 0, w_{kr}\geq 0, a_i\geq 0, b_j\geq 0, c_k\geq 0.\end{aligned} \tag{12}$$

其中 λ_1,λ_2 是正则化系数。

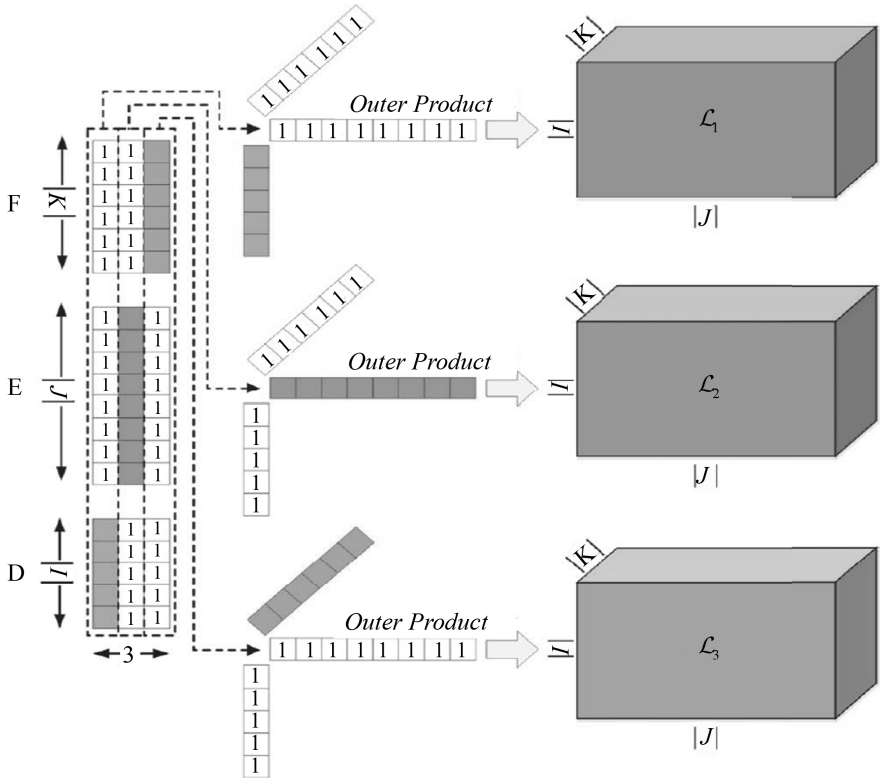


Figure 3. LBs establishment process
图 3. 线性偏置建立过程

3.2. 模型推导

根据(10)式，令 $A_{(t-1)},B_{(t-1)},C_{(t-1)},U_{(t-1)},V_{(t-1)}$ 和 $W_{(t-1)}$ 表示 LFs 中所要训练的参数第$(t-1)$次迭代之后的状态， $A'_{(t-1)},B'_{(t-1)},C'_{(t-1)},U'_{(t-1)},V'_{(t-1)}$ 和 $W'_{(t-1)}$ 表示第 t 次迭代时 A,B,C,U,V 和 W 的中间状态。在(10)式的基础上，当 t=1 时有：

$$\left[A'_{(0)},B'_{(0)},C'_{(0)},U'_{(0)},V'_{(0)},W'_{(0)}\right]^{\mathrm{T}}=\left[A_{(0)},B_{(0)},C_{(0)},U_{(0)},V_{(0)},W_{(0)}\right]^{\mathrm{T}} \tag{13}$$

其中， $A_{(0)},B_{(0)},C_{(0)},U_{(0)},V_{(0)},W_{(0)}$ 是随机产生的非负数，由(10)式可以得到第一次迭代之后的状态：

$$\left[A_{(1)},B_{(1)},C_{(1)},U_{(1)},V_{(1)},W_{(1)}\right]^{\mathrm{T}}\xleftarrow{\text{SLF-NMU}}\underset{A,B,C,U,V,W}{\operatorname{argmin}} Loss\left[A_{(0)},B_{(0)},C_{(0)},U_{(0)},V_{(0)},W_{(0)}\right]^{\mathrm{T}} \tag{14}$$

因此 $A'_{(1)},B'_{(1)},C'_{(1)},U'_{(1)},V'_{(1)},W'_{(1)}$ 的中间状态可以计算为：

$$\begin{aligned}&\left[A'_{(1)},B'_{(1)},C'_{(1)},U'_{(1)},V'_{(1)},W'_{(1)}\right]^{\mathrm{T}}=\left[A_{(1)},B_{(1)},C_{(1)},U_{(1)},V_{(1)},W_{(1)}\right]^{\mathrm{T}}\\&+\mu\left(\left[A_{(1)},B_{(1)},C_{(1)},U_{(1)},V_{(1)},W_{(1)}\right]^{\mathrm{T}}-\left[A_{(0)},B_{(0)},C_{(0)},U_{(0)},V_{(0)},W_{(0)}\right]^{\mathrm{T}}\right)\end{aligned}\tag{15}$$

其中 $A'_{(1)},B'_{(1)},C'_{(1)},U'_{(1)},V'_{(1)},W'_{(1)}$ 表示 A,B,C,U,V 和 W 第二次迭代时的中间状态。

值得注意是(15)式实际上是将 NAG 效应纳入学习过程中计算了 A,B,C,U,V 和 W 位置的偏移，但是由于有负项存在，不能保证 LFTs 中训练的参数非负性。

通过结合(10)和(15)可以得到：

$$\left[A_{(2)},B_{(2)},C_{(2)},U_{(2)},V_{(2)},W_{(2)}\right]^{\mathrm{T}}\xleftarrow{\text{SLF-NMU}}\underset{A,B,C,U,V,W}{\arg\min}\ Loss\left[A'_{(1)},B'_{(1)},C'_{(1)},U'_{(1)},V'_{(1)},W'_{(1)}\right]^{\mathrm{T}}\tag{16}$$

这表明在第二次迭代时，NAG 方法通过(9)式将 A,B,C,U,V 和 W 位置偏移纳入到学习过程中，然后根据(9)将 SLF-NMU 应用到最终目标中。类似地，可以得到第三到第 t 次迭代的更新规则。因此，结合(13)~(16)式得到以下训练方法：

$$\begin{cases}t=1: \begin{bmatrix}A_{(t)}\\B_{(t)}\\C_{(t)}\\U_{(t)}\\V_{(t)}\\W_{(t)}\end{bmatrix}\xleftarrow{\text{SLF-NMU}}\underset{A,B,C,U,V,W}{\arg\min}\ Loss\begin{bmatrix}A_{(t-1)}\\B_{(t-1)}\\C_{(t-1)}\\U_{(t-1)}\\V_{(t-1)}\\W_{(t-1)}\end{bmatrix}\\ t\geq 2: \begin{cases}\begin{bmatrix}A'_{(t-1)}\\B'_{(t-1)}\\C'_{(t-1)}\\U'_{(t-1)}\\V'_{(t-1)}\\W'_{(t-1)}\end{bmatrix}=\begin{bmatrix}A_{(t-1)}\\B_{(t-1)}\\C_{(t-1)}\\U_{(t-1)}\\V_{(t-1)}\\W_{(t-1)}\end{bmatrix}+\mu\left(\begin{bmatrix}A_{(t-1)}\\B_{(t-1)}\\C_{(t-1)}\\U_{(t-1)}\\V_{(t-1)}\\W_{(t-1)}\end{bmatrix}-\begin{bmatrix}A_{(t-2)}\\B_{(t-2)}\\C_{(t-2)}\\U_{(t-2)}\\V_{(t-2)}\\W_{(t-2)}\end{bmatrix}\right)\\ \begin{bmatrix}A_{(t-1)}\\B_{(t-1)}\\C_{(t-1)}\\U_{(t-1)}\\V_{(t-1)}\\W_{(t-1)}\end{bmatrix}\xleftarrow{\text{SLF-NMU}}\underset{A,B,C,U,V,W}{\arg\min}\ Loss\begin{bmatrix}A'_{(t-1)}\\B'_{(t-1)}\\C'_{(t-1)}\\U'_{(t-1)}\\V'_{(t-1)}\\W'_{(t-1)}\end{bmatrix}\end{cases}\end{cases}\tag{17}$$

通过结合(17)式，(12)式和 SLF-NMU 算法可以得到 A,B,C,U,V 和 W 学习过程的规则如下：

$$
\begin{cases}
t=1: \begin{cases}
a_{i(1)} \leftarrow \dfrac{a_{i(0)} \sum_{y_{ijk} \in \Lambda(i)} y_{ijk}}{\sum_{y_{ijk} \in \Lambda(i)} \hat{y}_{ijk(0)} + \lambda_2 \left|\Lambda(i)\right| a_{i(0)}} \\
b_{j(1)} \leftarrow \dfrac{b_{j(0)} \sum_{y_{ijk} \in \Lambda(j)} y_{ijk}}{\sum_{y_{ijk} \in \Lambda(j)} \hat{y}_{ijk(0)} + \lambda_2 \left|\Lambda(j)\right| b_{j(0)}} \\
c_{k(1)} \leftarrow \dfrac{c_{k(0)} \sum_{y_{ijk} \in \Lambda(k)} y_{ijk}}{\sum_{y_{ijk} \in \Lambda(k)} \hat{y}_{ijk(0)} + \lambda_2 \left|\Lambda(k)\right| c_{k(0)}} \\
u_{ir(1)} \leftarrow \dfrac{u_{ir(0)} \sum_{r_{ijk} \in \Lambda(i)} y_{ijk} \left(v_{jr(0)} + w_{kr(0)}\right)}{\sum_{r_{ijk} \in \Lambda(i)} \left(\hat{y}_{ijk(0)} \left(v_{jr(0)} + w_{kr(0)}\right)\right) + \lambda_1 \left|\Lambda(i)\right| u_{ir(0)}} \\
v_{jr(1)} \leftarrow \dfrac{v_{j,r(0)} \sum_{r_{ijk} \in \Lambda(j)} y_{ijk} \left(u_{ir(0)} + w_{kr(0)}\right)}{\sum_{r_{ijk} \in \Lambda(j)} \left(\hat{y}_{ijk(0)} \left(u_{ir(0)} + w_{kr(0)}\right)\right) + \lambda_1 \left|\Lambda(j)\right| v_{jr(0)}} \\
w_{kr(1)} \leftarrow \dfrac{w_{k,r(0)} \sum_{r_{ijk} \in \Lambda(k)} y_{ijk} \left(u_{ir(0)} + v_{jr(0)}\right)}{\sum_{r_{ijk} \in \Lambda(k)} \left(\hat{y}_{ijk(0)} \left(u_{ir(0)} + v_{jr(0)}\right)\right) + \lambda_1 \left|\Lambda(k)\right| w_{kr(0)}}
\end{cases} \\
t \geq 2: \begin{cases}
a'_{i(t-1)} \leftarrow a_{i(t-1)} + \max\left\{\mu\left(a_{i(t-1)} - a_{i(t-2)}\right), 0\right\} \\
b'_{j(t-1)} \leftarrow b_{j(t-1)} + \max\left\{\mu\left(b_{j(t-1)} - b_{j(t-2)}\right), 0\right\} \\
c'_{k(t-1)} \leftarrow c_{k(t-1)} + \max\left\{\mu\left(c_{k(t-1)} - c_{k(t-2)}\right), 0\right\} \\
u'_{ir(t-1)} \leftarrow u_{ir(t-1)} + \max\left\{\mu\left(u_{ir(t-1)} - u_{ir(t-2)}\right), 0\right\} \\
v'_{jr(t-1)} \leftarrow v_{jr(t-1)} + \max\left\{\mu\left(v_{jr(t-1)} - v_{jr(t-2)}\right), 0\right\} \\
w'_{kr(t-1)} \leftarrow w_{kr(t-1)} + \max\left\{\mu\left(w_{kr(t-1)} - w_{kr(t-2)}\right), 0\right\}
\end{cases} \\
t \geq 2: \begin{cases}
a_{i(t)} \leftarrow \dfrac{a'_{i(t-1)} \sum_{y_{ijk} \in \Lambda(i)} y_{ijk}}{\sum_{y_{ijk} \in \Lambda(i)} \hat{y}'_{ijk(t-1)} + \lambda_2 \left|\Lambda(i)\right| a'_{i(t-1)}} \\
b_{j(t)} \leftarrow \dfrac{b'_{j(t-1)} \sum_{y_{ijk} \in \Lambda(j)} y_{ijk}}{\sum_{y_{ijk} \in \Lambda(j)} \hat{y}'_{ijk(t-1)} + \lambda_2 \left|\Lambda(j)\right| b'_{j(t-1)}} \\
c_{k(t)} \leftarrow \dfrac{c'_{k(t-1)} \sum_{y_{ijk} \in \Lambda(k)} y_{ijk}}{\sum_{y_{ijk} \in \Lambda(k)} \hat{y}'_{ijk(t-1)} + \lambda_2 \left|\Lambda(k)\right| c'_{k(t-1)}} \\
u_{ir(t)} \leftarrow \dfrac{u'_{ir(t-1)} \sum_{r_{ijk} \in \Lambda(i)} y_{ijk} \left(v'_{jr(t-1)} + w'_{kr(t-1)}\right)}{\sum_{r_{ijk} \in \Lambda(i)} \left(\hat{y}'_{ijk(t-1)} \left(v'_{jr(t-1)} + w'_{kr(t-1)}\right)\right) + \lambda_1 \left|\Lambda(i)\right| u'_{ir(t-1)}} \\
v_{jr(t)} \leftarrow \dfrac{v'_{jr(t-1)} \sum_{r_{ijk} \in \Lambda(j)} y_{ijk} \left(u'_{ir(t-1)} + w'_{kr(t-1)}\right)}{\sum_{r_{ijk} \in \Lambda(j)} \left(\hat{y}'_{ijk(t-1)} \left(u'_{ir(t-1)} + w'_{kr(t-1)}\right)\right) + \lambda_1 \left|\Lambda(j)\right| v'_{jr(t-1)}} \\
w_{kr(t)} \leftarrow \dfrac{w'_{kr(t-1)} \sum_{r_{ijk} \in \Lambda(k)} y_{ijk} \left(v'_{jr(t-1)} + u'_{ir(t-1)}\right)}{\sum_{r_{ijk} \in \Lambda(k)} \left(\hat{y}'_{ijk(t-1)} \left(v'_{jr(t-1)} + u'_{ir(t-1)}\right)\right) + \lambda_1 \left|\Lambda(k)\right| w'_{kr(t-1)}}
\end{cases}
\end{cases}
\tag{18}
$$

其中 $\hat{y}_{ijk}$ 和 $\hat{y}'_{ijk}$ 计算方式如下所示：

$$\hat{y}_{ijk}=\sum_{r=1}^{f}\left(u_{ir}v_{jr}+u_{ir}w_{kr}+v_{jr}w_{kr}\right)+a_i+b_j+c_k \tag{19}$$

$$\hat{y}'_{ijk}=\sum_{r=1}^{f}\left(u'_{ir}v'_{jr}+u'_{ir}w'_{kr}+v'_{jr}w'_{kr}\right)+a'_i+b'_j+c'_k \tag{20}$$

从(18)式有如下发现：

1) 在广义化的 NAG 中存在 $\mu\left(a_{i(t-1)}-a_{i(t-2)}\right)$，$\mu\left(b_{j(t-1)}-b_{j(t-2)}\right)$，$\mu\left(c_{k(t-1)}-c_{k(t-2)}\right)$，$\mu\left(u_{ir(t-1)}-u_{ir(t-2)}\right)$，$\mu\left(v_{jr(t-1)}-v_{jr(t-2)}\right)$ 和 $\mu\left(w_{kr(t-1)}-w_{kr(t-2)}\right)$ 可能使训练过程中出现负项从而不能保证模型整体的非负性，所以采用了截断策略，始终保证速度项的非负性，从而保证模型整体的非负性；

2) 在第首次迭代时采用的是 SLF-NMU 方法来更新 A,B,C,U,V 和 W；

3) 从第二次迭代开始，(18)式不断的把 NAG 纳入学习过程中。

4. 超参数的自适应性

从(18)式可以看到广义化 NAG 算法依赖三个超参数，即正则化系数 λ_1,λ_2 和速度参数 μ，为了让参数可以自适应的调节大小，根据[14]采用粒子群优化(Particle Swarm Optimization, PSO)算法可以实现训练模型时超参数的自适应。根据这个原则首先建立一个由 p 个粒子组成的搜索空间。其中，第 j 个粒子的位置和速度定义分别为：$q_j=\left(\lambda_j,\mu_j\right)$ 和 $v_j=\left(v_{j\lambda},v_{j\mu}\right)$，则第 j 个粒子在第 t 次迭代的公式为：

$$v_{j(t)}=\omega v_{j(t-1)}+c_1r_1\left(pb_{j(t-1)}-q_{j(t-1)}\right)+c_2r_2\left(gb_{(t-1)}-q_{j(t-1)}\right) \tag{21}$$

$$q_{j(t)}=q_{j(t-1)}+v_{j(t)}$$

其中 $pb_{j(t-1)}$ 表示第 j 个粒子发现的最佳位置，$gb_{(t-1)}$ 表示第 t 次迭代中整个粒子群发现的最佳位置。(21)式采用的是标准的 PSO [14]，兼容表 1 中 PSO 常用的参数。为了简化下面实验这里令 $\lambda=\lambda_1=\lambda_2$。根据[14]限制 v_λ,v_μ,λ 和 μ 在如下的搜索空间：$v_\lambda\in\left[\breve{v}_\lambda,\tilde{v}_\lambda\right]$，$v_\mu\in\left[\breve{v}_\mu,\tilde{v}_\mu\right]$，$\lambda\in\left[\breve{\lambda},\tilde{\lambda}\right]$，$\mu\in\left[\breve{\mu},\tilde{\mu}\right]$。根据实验经验，本文采用表 2 中所给的数据。

Table 1. PSO parameter setting
表 1. PSO 参数设置

ω	c_1	c_2	r_1,r_2
0.724	2	2	$r_1,r_2\in[0,1]$ 随机数

Table 2. Parameter setting
表 2. 参数设置

$\breve{\lambda}$	$\tilde{\lambda}$	$\breve{\mu}$	$\tilde{\mu}$	$\breve{v}_\lambda$	$\tilde{v}_\lambda$	$\breve{v}_\mu$	$\tilde{v}_\mu$
0.01	0.1	0.8	1.4	−0.018	0.018	−0.12	0.12

由于本章的目标是在一个 HiDS 张量上实现高度精确的缺失数据估计，因此将第 j 个粒子的适应度函数定义为：

$$F(j)=\sqrt{\sum_{y_{ijk}\in\Gamma}\left(y_{ijk}-\hat{y}_{ijk}\right)^2\Big/|\Gamma|} \tag{22}$$

其中$|\cdot|$表示计算集合中元素个数，Γ 是测试集并且 $\Gamma\cap\Lambda=\varnothing$，$\hat{y}_{ijk}$ 表示有模型估计出的值，y_{ijk} 表示 HiDS 张量中的真实值。由(21)式、表 1 和表 2 可以实现 λ 和 μ 的自适应。

5. NHLFT-NAG 算法设计与分析

基于以上讨论，设计出 NHLFT-NAG 模型的算法如算法 1 所示。为了增加算法运行的效率在算法中引入了几个辅助矩阵。例如 U 依赖于 $U^{(U)},U^{(D)},U^{(O)}$ 和 $U^{(H)}$。其中 $U^{(U)}$ 和 $U^{(D)}$ 表示存储每一次迭代中的学习增量，$U^{(O)}$ 和 $U^{(H)}$ 表示存储 U 最后两次迭代的中间状态。为了使 λ_1,λ_2 和 μ 达到自适应的效果，引入辅助矩阵 P,V,P_B 和 F 分别存储了更新的位置，更新速度，最好的位置和粒子的适应度的函数值。

算法 1. NHLFT-NAG 算法

输入： $Y,\Lambda,f,\omega,c_1,c_2$

输出： U,V,W,A,B,C

1. 初始化：U,V,W,A,B,C 及相应的辅助矩阵；PSO 相关的参数，最大迭代次数 T，收敛阈值 δ
2. 根据(12)式计算损失函数值
3. 根据(18)式计算 U,V,W,A,B,C
4. $U^{(H)}=U^{(O)},U^{(O)}=U,V^{(H)}=V^{(O)},V^{(O)}=V^{(O)}W^{(H)}=W^{(O)},\quad W^{(O)}=W,A^{(H)}=A^{(O)},A^{(O)}=AB^{(H)}=B^{(O)},B^{(O)}=B,$ $C^{(H)}=C^{(O)},C^{(O)}=C$
5. 计算适应度函数 F_p 值
6. 利用 PSO 进行参数优化
7. 目标函数收敛或者达到最大迭代次数，循环结束，否则返回 3

NHLFT-NAG 模型的存储耗费依赖于 $\Lambda,I,J,K,U,V,W,A,B,C$ 和相关的辅助矩阵，所以存储耗费为：

$$\begin{aligned}S&=\Theta\left(|\Lambda|+5\times\left(|I|+|J|+|K|\right)\times f+6\times\left(|I|+|J|+|K|\right)+7\times p\right)\\&\approx\Theta\left(|\Lambda|+5\times\left(|I|+|J|+|K|\right)\times f\right)\end{aligned}\tag{23}$$

从(23)式可以看出由于存储耗费和 $|\Lambda|,|I|,|J|,|K|$ 线性相关所以在现实中容易实施。

6. 实验结果与分析

6.1. 数据集

实验采用的两个数据集是由 WSMonitor 收集的[7]，相关细节由表 3 给出。从表 3 可以看到，两个数据集分别描述了吞吐量和响应时间，记录了 142 个用户在 64 个不同的时间点对 4532 个真实的网络服务信息，每个数据集拥有 30,287,611 个 Qos 记录。实际上可以得到两个规模为 142 × 4532 × 64 的用户－服务－时间的 QoS 张量，每个数据集的密度为 73.53%。

Table 3. Dataset description
表 3. 数据集描述

数据集	D1	D2
数据类型	响应时间	吞吐量
数据范围	0~20 s	0~1000 kbps

Continued

均值	3.165 s	9.609 kbps
用户数量	142	142
服务数量	4532	4532
时间点数量	64	64
总数量	30,287,611	30,287,611

为了易于比较，在实验之前将 D1 和 D2 两个数据集的数据映射到[0, 5]。将两个数据集随机的划分成 70%训练集 Λ，20%测试集 Γ，10%验证集 K。为了消除偏差和主观因素的影响采用 10 折交叉验证，训练过程将被重复 50 次。训练迭代停止时满足以下条件之一：1) 当误差达到 10^{-5}；2) 达到预先设置的最大迭代次数 1000 次。

6.2. 评估指标

本文关注模型生成数据的准确性，因为它直接反映模型能否精确的挖掘 HiDS 张量的基本特征，故采用平均绝对误差(Mean Absolute Error, MAE)和均方根误差(Root Mean Square Error, RMSE)作为评价标准。

$$\mathrm{MAE} = \sum_{y_{ijk}\in\Psi} \left|y_{ijk} - \hat{y}_{ijk}\right| \Big/ |\Psi| \tag{24}$$

$$\mathrm{RMSE} = \sqrt{\sum_{y_{ijk}\in\Psi} \left(y_{ijk} - \hat{y}_{ijk}\right)^2 \Big/ \|\Psi\|} \tag{25}$$

其中 Ψ 表示验证集，并且 $\Psi \cap \Lambda = \varnothing$，$\hat{y}_{ijk}$ 表示测试的项目来自测试集的预测。对于一个模型来说， MAE 和 RMSE 的值越小代表模型预测精度越高。

6.3. 对比实验和参数设置

选取了五个模型作对比实验相关细节由表 4 给出。

Table 4. Contrast model
表 4. 对比模型

模型	描述
M1	NHLFT 模型采用 SGD [15]学习超参数
M2	NHLFT 模型采用 SLF-NMU [7]学习超参数
M3	NHLFT 模型采用动量法[10]学习超参数
M4	不带偏置项的 NHLFT-NAG 模型
M5	带有偏置项的 NHLFT-NAG 模型

M1~M5 都是线性模型其预测的精度依赖潜在因子的维数 f，统一设置 $f = 20$ 来平衡预测精度和计算效率。模型 M1 依赖正则化系数 λ 和学习率 η，由表 5 给出。

Table 5. M1's hyperparameter settings on D1, D2
表 5. M1 在 D1，D2 上超参数的设置

数据集	D1	D2
λ	0.02	0.062
η	0.0002	0.0004

模型 M2 依赖正则化系数 λ，由表 6 给出。

Table 6. M2's hyperparameter settings on D1, D2
表 6. M2 在 D1，D2 上超参数的设置

数据集	D1	D2
λ	0.041	0.054

模型 M3 依赖正则化系数 λ 和加速度常数 μ，由表 7 给出。

Table 7. M3's hyperparameter settings on D1, D2
表 7. M3 在 D1，D2 上超参数的设置

数据集	D1	D2
λ	0.042	0.057
μ	0.85	1.32

6.4. 实验结果

表 8，表 9 和表 10 分别记录了五个模型最好的 RMSE，MAE 和训练时间。

Table 8. RMSE of M1~M5 on D1, D2
表 8. M1~M5 在 D1，D2 上的 RMSE

数据集	A1	A2	A3	A4	A5
D1	0.9799	0.7886	0.7954	0.7916	0.7437
D2	0.6352	0.5046	0.5204	0.4945	0.4671

Table 9. MAE of M1~M5 on D1, D2
表 9. M1~M5 在 D1，D2 上的 MAE

数据集	A1	A2	A3	A4	A5
D1	0.4887	0.4475	0.4518	0.4404	0.4012
D2	0.4251	0.3136	0.3289	0.3202	0.2938

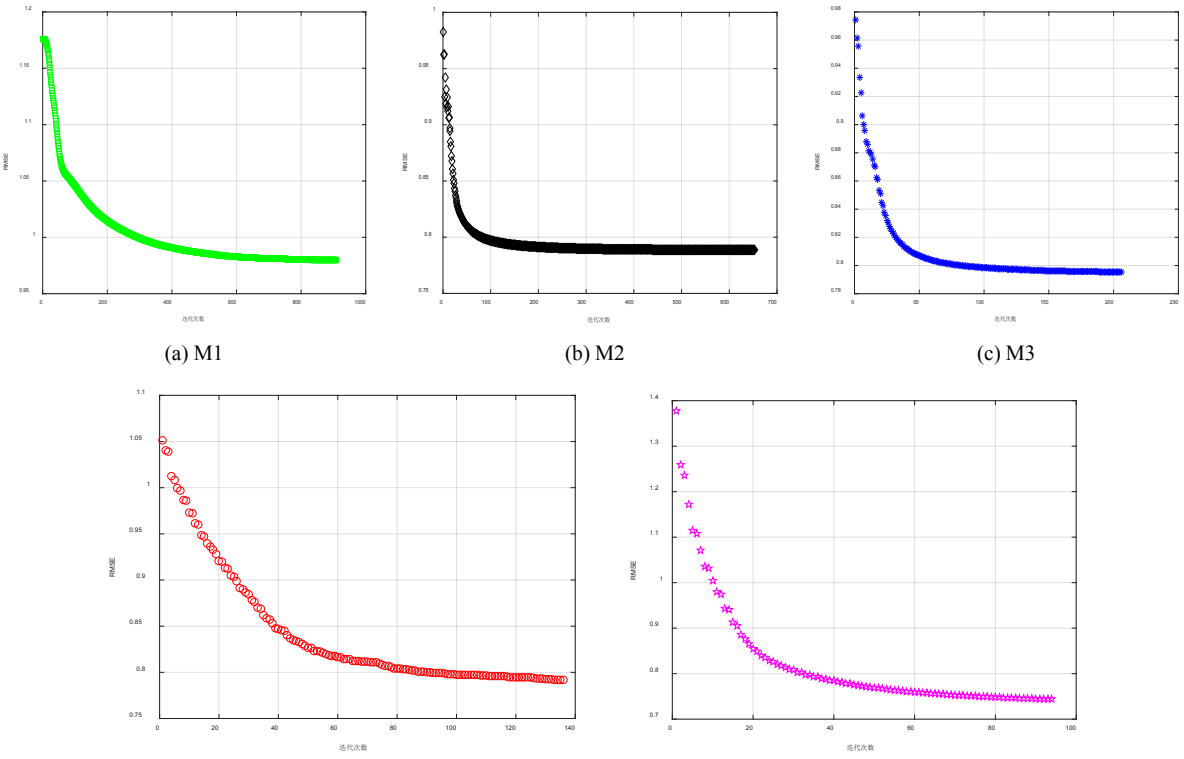
Table 10. Training time of M1~M5 on D1, D2 (second)
表 10. M1~M5 在 D1，D2 上训练时间(秒)

数据集	A1	A2	A3	A4	A5
D1	4936	3672	1789	562	573
D2	5724	4597	1962	659	668

从表 8 可以看出在 D1 上 M5 的 RMSE 为 0.7437 相比于 M1 的 0.9799 降低了 24.1%，相比于 M2 的 0.7886 降低了 5.69%，相比于 M3 的 0.7954 降低了 6.49%，相比于 M4 的 0.7916 降低了 6.05%。在 D2 上 M5 的 RMSE 为 0.4671 相比于 M1 的 0.6352 降低了 26.4%，相比于 M2 的 0.5046 降低了 7.43%，相比于 M3 的 0.5204 降低了 10.2%，相比于 M4 的 0.4945 降低了 5.54%。M4 和 M5 对 HiDS 张量缺失数据的预测精度相比于 M1~M3 要高。其中 M5 由于对用户，服务，时间分别增加了线性偏置，使得模型的预测精度有了进一步提升。类似的结论也可以从表 9 得到。

从表 10 可以看出 M5 的计算效率明显高于它的同类型的算法。例如，在 D1 上 M5 的训练时间为 573 秒，相比于 M1 的 4936 秒降低了 88.3%，相比于 M2 的 3672 秒降低了 84.3%，相比于 M3 的 1789 秒降低了 67.9%。而相比于 M4 的 562 秒，M5 的训练时间稍长，因为在 M5 中引入了线性偏置，引入了更多的自由变量，使得模型训练的参数变多因此时间增加。超参数的调节往往是要耗费大量的时间，而表 10 记录的是 M1~M3 预先调好参数之后模型的训练时间，本文提出的 M5 超参数时通过 PSO 自适应的调节，从而节约了大量的时间。

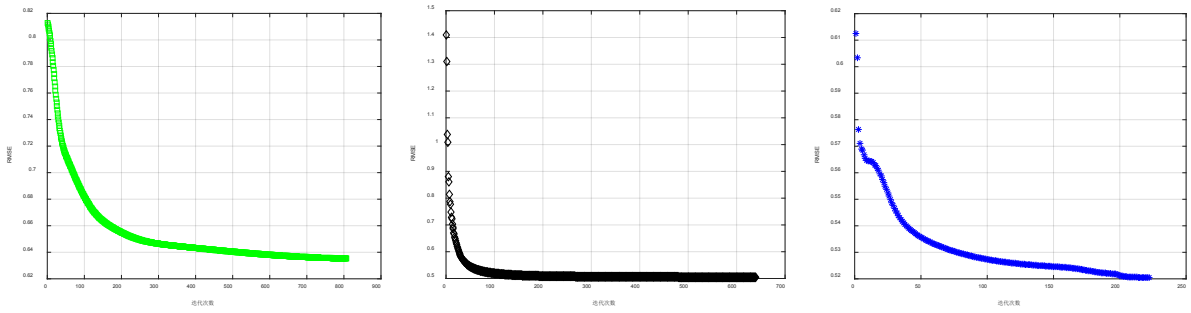
图 4~7 分别记录了 M1~M5 在 D1 和 D2 上训练时的 RMSE 和 MAE 的收敛曲线。从图 4 和图 5 中可以看出当 NHLFT 模型采用了广义化 NAG 算法和参数的自适应调节之后模型的收敛速度明显高于同类型的算法(M1, M2, M3)，例如在 D1 上 M5 (图 4(e))只迭代了 94 次就已经收敛而 M1，M2，M3 分别迭代了 903，653，206 次才收敛，这个结果明显优于同类型的算法。注意当加入 PSO 调节超参数的自适应之后 NHLFT-NAG 模型每次迭代都要遍历所有的粒子，因此每次迭代实际上是由 p 个子迭代构成。但是即使是这样 M5 的收敛速度依然是最快的。类似的结论在图 6 和图 7 中得到。



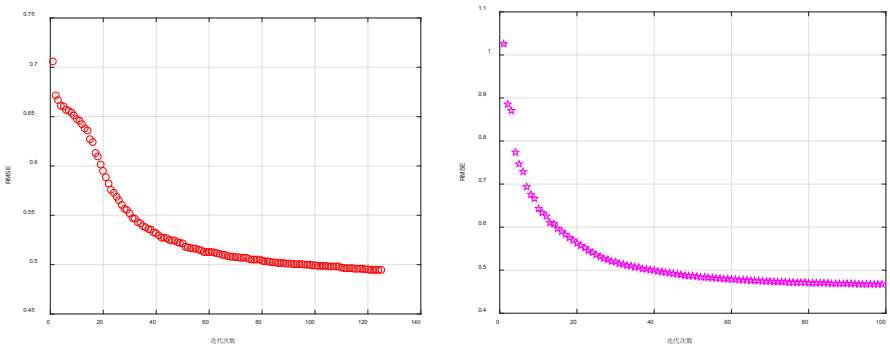
(a) M1　(b) M2　(c) M3

(d) M4　(e) M5

Figure 4. RMSEs of M1~M5 on D1
图 4. M1~M5 在 D1 上的 RMSEs

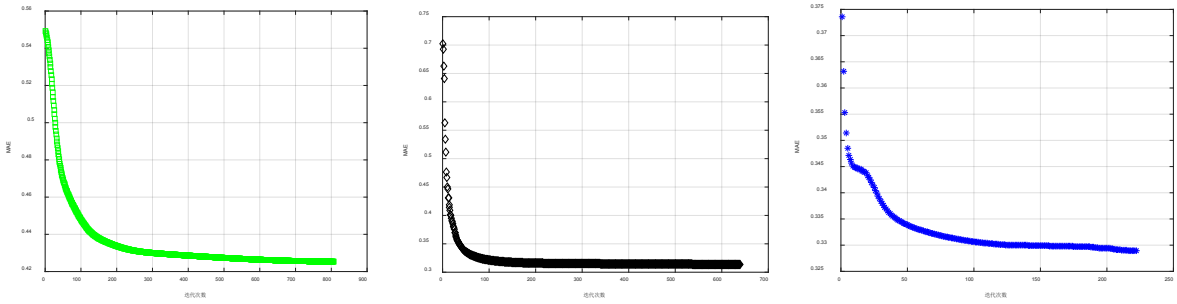


(a) M1 (b) M2 (c) M3

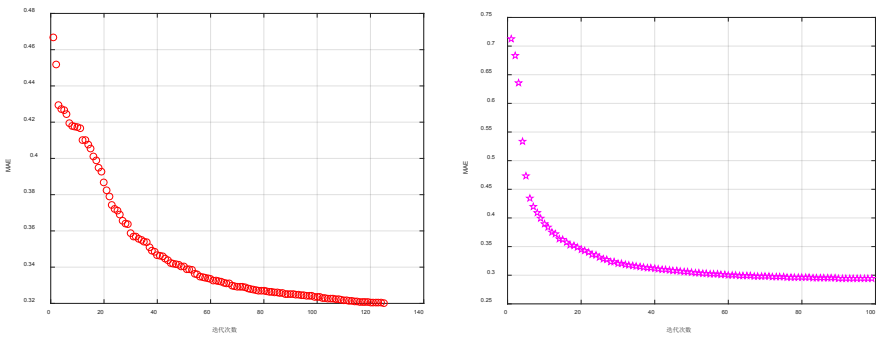


(d) M4 (e) M5

Figure 5. RMSEs of M1~M5 on D2
图 5. M1~M5 在 D2 上的 RMSEs

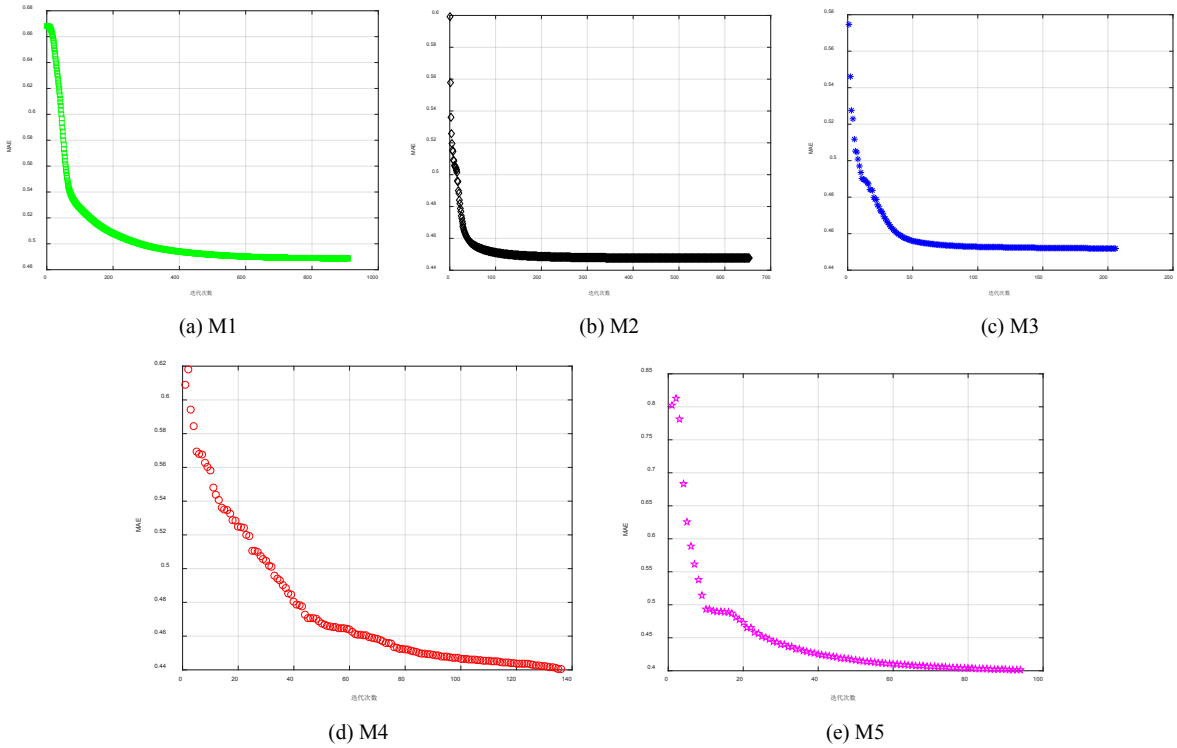


(a) M1 (b) M2 (c) M3



(d) M4 (e) M5

Figure 6. MAEs of M1~M5 on D1
图 6. M1~M5 在 D1 上的 MAEs



(a) M1 (b) M2 (c) M3

(d) M4 (e) M5

Figure 7. MAEs of M1~M5 on D2
图 7. M1~M5 在 D2 上的 MAEs

7. 结论

本章主要目的是预测动态 QoS 数据中的缺失值，将动态数据建模成用户－服务－时间的张量，进行潜在因子张量分解，为此提出了 NHLFT-NAG 模型，该模型考虑了：1) 线性偏置；2) 数据的非负性；3) 超参数的自适应性。因此本文提出的 NHLFT-NAG 模型不仅加快了收敛速度而且在精度上也取得了一定的优势。将来可以对 NHLFT-NAG 模型进行进一步拓展，如改变模型的正则化项，此外还可以考虑采用其他优化算法来对模型的超参数进行优化。

基金项目

多性能指标下随机 Markov 跳变系统的预测控制及其在风洞流场中的应用，国家自然科学基金资助(62073223)；基于通讯协议的复杂网络同步，上海市自然科学基金资助(18ZR1427100)。

参考文献

[1] Ghahramani, M.H., Zhou, M.C. and Hon, C.T. (2017) Toward Cloud Computing QoS Architecture: Analysis of Cloud Systems and Cloud Services. *IEEE/CAA Journal of Automatica Sinica*, **4**, 6-18. https://doi.org/10.1109/JAS.2017.7510313

[2] Fareghzadeh, N. (2022) An Architecture Supervisor Scheme toward Performance Differentiation and Optimization in Cloud Systems. *The Journal of Supercomputing*, **78**, 1532-1563. https://doi.org/10.1007/s11227-021-03846-w

[3] Luo, X., Zhou, M.C., Xia, Y., *et al*. (2015) Generating Highly Accurate Predictions for Missing QoS Data via Aggregating Nonnegative Latent Factor Models. *IEEE Transactions on Neural Networks and Learning Systems*, **27**, 524-537. https://doi.org/10.1109/TNNLS.2015.2412037

[4] Luo, X., Zhou, M.C., Li, S., *et al*. (2017) Incorporation of Efficient Second-Order Solvers into Latent Factor Models for Accurate Prediction of Missing QoS Data. *IEEE Transactions on Cybernetics*, **48**, 1216-1228.

https://doi.org/10.1109/TCYB.2017.2685521

[5] Chang, Z., Ding, D. and Xia, Y. (2021) A Graph-Based QoS Prediction Approach for Web Service Recommendation. *Applied Intelligence*, **51**, 6728-6742. https://doi.org/10.1007/s10489-020-02120-5

[6] Koren, Y. (2010) Collaborative Filtering with Temporal Dynamics. *Communications of the ACM*, **53**, 89-97. https://doi.org/10.1145/1721654.1721677

[7] Luo, X., Wu, H., Yuan, H., *et al*. (2019) Temporal Pattern-Aware QoS Prediction via Biased Non-Negative Latent Factorization of Tensors. *IEEE Transactions on Cybernetics*, **50**, 1798-1809. https://doi.org/10.1109/TCYB.2019.2903736

[8] Nesterov, Y.E. (1983) A Method of Solving a Convex Programming Problem with Convergence Rate O (1/k2). *Soviet Mathematics Doklady*, **27**, 372-376.

[9] Attouch, H., Bolte, J. and Svaiter, B.F. (2013) Convergence of Descent Methods for Semi-Algebraic and Tame Problems: Proximal Algorithms, Forward-Backward Splitting, and Regularized Gauss-Seidel Methods. *Mathematical Programming*, **137**, 91-129. https://doi.org/10.1007/s10107-011-0484-9

[10] Luo, X., Liu, Z., Li, S., *et al*. (2018) A Fast Non-Negative Latent Factor Model Based on Generalized Momentum Method. *IEEE Transactions on Systems*, *Man*, *and Cybernetics*: *Systems*, **51**, 610-620. https://doi.org/10.1109/TSMC.2018.2875452

[11] Sutskever, I., Martens, J., Dahl, G., *et al*. (2013) On the Importance of Initialization and Momentum in Deep Learning. *International Conference on Machine Learning*, Atlanta, 16-21 June 2013, 1139-1147.

[12] Aggarwal, C.C. (2016) Recommender Systems. Springer International Publishing, Cham. https://doi.org/10.1007/978-3-319-29659-3

[13] Javed, K., Gouriveau, R. and Zerhouni, N. (2015) A New Multivariate Approach for Prognostics Based on Extreme Learning Machine and Fuzzy Clustering. *IEEE Transactions on Cybernetics*, **45**, 2626-2639. https://doi.org/10.1109/TCYB.2014.2378056

[14] Shi, Y. (2001) Particle Swarm Optimization: Developments, Applications and Resources. *Proceedings of the* 2001 *Congress on Evolutionary Computation*, Seoul, 27-30 May 2001, 81-86.

[15] Wijnhoven, R.G.J. and de With, P.H.N. (2010) Fast Training of Object Detection Using Stochastic Gradient Descent. 2010 20*th International Conference on Pattern Recognition*, Istanbul, 23-26 August 2010, 424-427. https://doi.org/10.1109/ICPR.2010.112