American Journal of Biochemistry and Biotechnology

Original Research Paper

A Soft Sensor Modelling of Biomass Concentration during Fermentation using Accurate Incremental Online ν-Support Vector Regression Learning Algorithm

Binjie Gu and Feng Pan

Key Laboratory of Advanced Process Control for Light Industry, Ministry of Education, Jiangnan University, Wuxi, China

Article history
Received: 28-07-2015
Revised: 01-08-2015
Accepted: 03-08-2015

Corresponding Author:
Binjie Gu
Key Laboratory of Advanced Process Control for Light Industry, Ministry of Education, Jiangnan University, Wuxi, China
Email: gubinjie1980@126.com

Abstract: In order to model real fermentation process, a soft sensor modelling of biomass concentration during fermentation using accurate incremental online ν-Support Vector Regression (ν-SVR) learning algorithm was proposed. Firstly, an accurate incremental online ν-SVR learning algorithm was proposed. This algorithm solved the two complications introduced in the dual problem based on the equivalent formulation of ν-SVR. Moreover, it addressed the infeasible updating path problem during the adiabatic incremental process by relaxed adiabatic incremental adjustments and accurate incremental adjustments. Then, the proposed algorithm is used to predict the biomass concentration of glutamic acid fed-batch fermentation process online. The results of simulation experiment showed that the soft sensor modelling of biomass concentration during fermentation using the proposed algorithm was of better generalization ability and cost less training time than that of ν-SVR.

Keywords: Soft Sensor Modelling, ν-Support Vector Regression, Online Learning, Biomass Concentration, Fed-Batch Fermentation

Introduction

The optimal control of fermentation process plays an important role in microbial fermentation, which concerns the success or failure of industrial production (Wei and Yang, 2008). According to different process flow, fermentation process can be partitioned into three categories: Batch fermentation, continuous fermentation and fed-batch fermentation (Shi and Pan, 2010). Fed-batch fermentation has successfully solved problems such as substrate inhibition, strain degeneration and contamination existed in batch fermentation or continuous fermentation. Moreover, the optimal control of fermentation process can be realized easily. Therefore, fed-batch fermentation has been widely used in industrial fermentation in recent years.

It is well known that fermentation process is time-variant and nonlinear (Chen and Li, 2002; Yu, 2011). Therefore, in order to realize the optimal control of fermentation process, it is necessary to master the state information of fermentation process timely. Unfortunately, some important variables, including biomass concentration, product concentration or substrate concentration are rather difficult to be measured online in real fermentation process. This seriously influences the implementation of optimization strategy (Yoshida *et al*., 1973). To address this issue, the most widely used approach is soft sensor modelling based on training samples. The idea of soft sensor modelling is to infer or estimate some important variables which cannot or rather difficult to be measured by establishing mathematical relations based on other known or easily measurable variables. Therefore, to some extent, the soft sensor modelling can replace the function of hardware.

Nowadays, many offline soft sensor modelling algorithms had been proposed and achieved remarkable results. Petrova *et al*. (1998) realized the modelling of biomass growth based on Artificial Neural Network (ANN), the chief drawback of this method was that large amount of samples were required to train ANN, which was unsuitable for small sample learning scenarios. Feng *et al*. (2004) proposed a soft sensor model of the Box-Jenkins gas furnace and FCCU based on weighted Support Vector Machine (SVM). Gao *et al*. (2006) proposed a soft sensor model of the penicillin fermentation process based on SVM. However, once the offline model was established, if the model parameters need to be tuned, we must wait until the whole production process was over, which caused re-training

the model from scratch. Therefore, they were unsuitable for real production process.

To overcome this problem, Wang *et al*. (2009) proposed a soft sensor modelling of biotechnical process based on online ε-Support Vector Regression (ε-SVR) and achieved better results than those of offline algorithms. Compared with offline algorithms, the model parameters of online algorithms can be tuned online, which is more suitable for real fermentation process. Unfortunately, it is rather difficult to select an appropriate C. To address this issue, Schölkopf *et al*. (2000) proposed the ν-SVR, which uses a new parameter ν to replace the parameter C. Moreover, it is easier to tune parameter ν than C. However, compared with the dual problem of ε-SVR, two complications are introduced in ν-SVR. The first one is that the box constraints are related to C and the length of the training samples and the second one is that one more inequality constraint is introduced (Gu *et al*., 2015). Moreover, as proved in Gu *et al*. (2012), it will not guarantee that a feasible updating path can always be generated. To sum up, it is rather difficult to design an online ν-SVR learning algorithm.

In this study, an accurate incremental online ν-SVR learning algorithm is proposed to address the problems mentioned above. Then, the proposed algorithm is applied in the soft sensor modelling of glutamic acid fed-batch fermentation process. Finally, compared with ν-SVR, the results of modelling are analyzed to show the superiority of the proposed algorithm.

Accurate Incremental Online ν-SVR Learning

To make the symbols easier to follow, we give a summary of the symbols at the end of this article.

Equivalent Formulation of ν-SVR

Given a training sample set $F=\{(x_1,y_1),\cdots,(x_l,y_l)\}$, such that $x_i\in \mathrm{R}^d$ is an input and $y_i\in$R, i = 1,…,l is a target output, the primal problem of ν-SVR is (Chang and Lin, 2002):

$$\begin{aligned}\min_{w,b,\varepsilon,\xi_i,\xi_i^*} &= \frac{1}{2}w^Tw+C\left(\nu\varepsilon+\frac{1}{l}\sum_{i=1}^{l}\left(\xi_i+\xi_i^*\right)\right)\\ s.t.\ &\left(w^T\phi(x_i)+b\right)-y_i\le\varepsilon+\xi_i\\ &y_i-\left(w^T\phi(x_i)+b\right)\le\varepsilon+\xi_i^*\\ &\xi_i,\xi_i^*\ge 0,i=1,\cdots,l,\varepsilon\ge 0\end{aligned}\tag{1}$$

Where:

w = The weight column vector
b = Bias
ξ_i and ξ_i^* = Nonnegative slack variables
l = The length of training samples

The training vectors x_i are mapped into a high dimensional Reproducing Kernel Hilbert Space (RKHS) by the transformation function ϕ. The ε-insensitive loss function means that if $w^T\phi(x_i)+b$ is in the range of $y_i\pm\varepsilon$, no loss is considered. ν is the introduced new proportion parameter with $0\le\nu\le 1$, which lets one control the number of support vectors and errors. To be more precise, ν is an upper bound on the fraction of margin errors and a lower bound of the fraction of support vectors. In addition, with probability 1, asymptotically, ν equals to both fractions (Gu *et al*., 2015). Therefore, it is easier to tune parameter ν than ε-SVR.

The dual problem of Equation 1 is (Chang and Lin, 2002):

$$\begin{aligned}\min_{\alpha_i,\alpha_i^*}\ & D=\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\left(\alpha_i-\alpha_i^*\right)H_{ij}\left(\alpha_j-\alpha_j^*\right)+\sum_{i=1}^{l}\left(\alpha_i-\alpha_i^*\right)y_i\\ s.t.\ & \sum_{i=1}^{l}\left(\alpha_i-\alpha_i^*\right)=0,\sum_{i=1}^{l}\left(\alpha_i+\alpha_i^*\right)\le C\nu,\\ & 0\le\alpha_i,\alpha_i^*\le\frac{C}{l},i=1,\cdots,l\end{aligned}\tag{2}$$

where, $H_{ij}=K(\boldsymbol{x}_i,\boldsymbol{x}_j)=\phi^T(x_i)\phi(x_j)$; K is the kernel function.

Compared with the dual problem of stand ε-SVR (Chang and Lin, 2002), it is clear that two complications are introduced in the dual problem of ν-SVR. The first one is that the box constraints $0\le\alpha_i,\alpha_i^*\le C/l$ are related to C and l, the second one is that Equation 2 has an extra inequality constraint.

To solve the first complication, we multiply the objective function P in Equation 1 by the length of training samples and consider the following primal problem:

$$\begin{aligned}\min_{w,b,\varepsilon,\xi_i,\xi_i^*} & P=\frac{l}{2}w^Tw+C\left(\nu l\varepsilon+\sum_{i=1}^{l}\left(\xi_i+\xi_i^*\right)\right)\\ s.t.\ & \left(w^T\phi(x_i)+b\right)-y_i\le\varepsilon+\xi_i\\ & y_i-\left(w^T\phi(x_i)+b\right)\le\varepsilon+\xi_i^*\\ & \xi_i,\xi_i^*\ge 0,i=1,\cdots,l,\varepsilon\ge 0\end{aligned}\tag{3}$$

It is easy to verify that Equation 3 is equivalent to Equation 1. The corresponding dual problem of Equation 3 is:

$$\begin{aligned}\min_{\alpha_i,\alpha_i^*}\ & D=\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\left(\alpha_i-\alpha_i^*\right)P_{ij}\left(\alpha_j-\alpha_j^*\right)+\sum_{i=1}^{l}\left(\alpha_i-\alpha_i^*\right)y_i'\\ s.t.\ & \sum_{i=1}^{l}\left(\alpha_i-\alpha_i^*\right)=0,\sum_{i=1}^{l}\left(\alpha_i+\alpha_i^*\right)\le\nu l\\ & 0\le\alpha_i,\alpha_i^*\le 1,i=1,\cdots,l\end{aligned}\tag{4}$$

where, $P_{ij} = H_{ij}/l$ and $y_i' = y_i/C$.

Furthermore, we can solve the second complication based on Lemma 1.

Lemma 1 For any given ν in Equation 4, if 0≤ν≤1, there are always optimal solutions which happen at the equality $\sum_{i=1}^{l}\left(\alpha_i + \alpha_i^*\right) = \nu l$.

The detailed proof of Lemma 1 can be found in Chang and Lin (2002), it is omitted here.

Based on Lemma 1, we consider the following dual problem instead of Equation 4:

$$\begin{aligned}&\min_{\alpha_i,\alpha_i^*} D = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\left(\alpha_i - \alpha_i^*\right)P_{ij}\left(\alpha_j - \alpha_j^*\right) + \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right)y_i' \\ &s.t. \quad \sum_{i=1}^{l}\left(\alpha_i - \alpha_i^*\right) = 0, \sum_{i=1}^{l}\left(\alpha_i + \alpha_i^*\right) = \nu l \\ &\quad 0 \le \alpha_i, \alpha_i^* \le 1, i = 1, \cdots, l\end{aligned} \tag{5}$$

In order to make Equation 5 more compact, let z_i denotes the label of the training sample (x_i, y_i'), we define the expanded training sample set T, which is defined as:

$$T = T^+ \cup T^- = \{(x_i, y_i', z_i = +1|_{i=1}^{l})\} \cup \{(x_i, y_i', z_i = -1)|_{i=1}^{l}\}$$

Then, Equation 5 can be rewritten as:

$$\begin{aligned}&\min_{\alpha_i} D = \frac{1}{2}\sum_{i=1}^{2l}\sum_{j=1}^{2l}\alpha_i Q_{ij}\alpha_j + \sum_{i=1}^{2l} z_i\alpha_i y_i' \\ &s.t. \quad \sum_{i=1}^{2l} z_i\alpha_i = 0, \sum_{i=1}^{2l}\alpha_i = \nu l, \\ &\quad 0 \le \alpha_i \le 1, i = 1, \cdots, 2l\end{aligned} \tag{6}$$

where, $Q_{ij} = z_i z_j \phi^T(x_i)\phi(x_j)/l$.

According to the convex optimization theory (Bertsekas, 2009), the solution of Equation 6 can be obtained by minimizing the following convex quadratic objective function under constraints:

$$\begin{aligned}\min_{0\le\alpha_i\le 1} W_\nu = &\frac{1}{2}\sum_{i=1}^{2l}\sum_{j=1}^{2l}\alpha_i Q_{ij}\alpha_j + \sum_{i=1}^{2l} z_i\alpha_i y_i' \\ &+ b\left(\sum_{i=1}^{2l} z_i\alpha_i\right) + \rho\left(\sum_{i=1}^{2l}\alpha_i - \nu l\right)\end{aligned} \tag{7}$$

where, b and ρ are Lagrange multipliers.

Optimizing Equation 7 leads to the following Karush_Kuhn_Tucker (KKT) conditions (Karush, 1939):

$$\begin{aligned}g_i = \frac{\partial W_\nu}{\partial \alpha_i} &= \sum_{j=1}^{2l}\alpha_j Q_{ij} + z_i y_i' + z_i b + \rho \\ &= \begin{cases} > 0 & \alpha_i = 0 \quad \forall i \in R \\ = 0 & 0 < \alpha_i < 1 \quad \forall i \in S \\ < 0 & \alpha_i = 1 \quad \forall i \in E \end{cases}\end{aligned} \tag{8}$$

$$\frac{\partial W_\nu}{\partial b} = 0 \Rightarrow \sum_{i=1}^{2l} z_i\alpha_i = 0 \tag{9}$$

$$\frac{\partial W_\nu}{\partial \rho} = 0 \Rightarrow \sum_{i=1}^{2l}\alpha_i = \nu l \tag{10}$$

Based on the value of g_i, the expanded training sample set T can be partitioned into three independent sets, which is shown in Fig. 1.

(a) The set S including margin support vectors strictly on the margin, for convenience, the numbers of training data points in the set S is donated as r; (b) The set E including error support vectors exceeding the margin; (c) The set R including the remaining vectors covered by the margin.

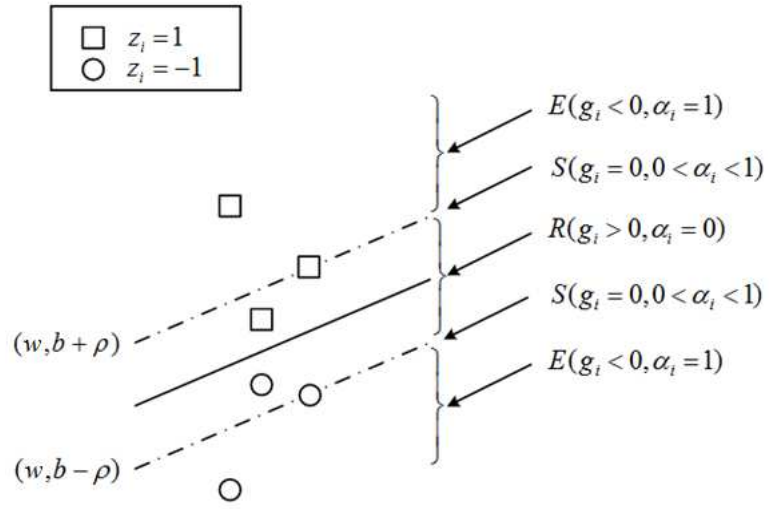

Fig. 1. The partition of the expanded training sample set T into three independent sets: (a) the set S; (b) the set E; (c) the set R

Accurate Incremental Online ν-SVR Learning Algorithm

The first accurate incremental online ε-support vector learning algorithm was proposed by Cauwenberghs and Poggio (2001) (further referred to as a C&P algorithm). Unfortunately, as proved in Gu *et al*. (2012), when the C&P algorithm is directly applied to the equivalent formulation of ν-SVR, it will not guarantee that a feasible updating path can always be generated. To address this issue, an accurate incremental online ν-SVR learning algorithm (it is called AIOSVR learning algorithm) is proposed in this section.

AIOSVR Learning Algorithm:

The AIOSVR learning algorithm is presented in Algorithm1.

Algorithm 1 The AIOSVR learning algorithm: High-level summary.

Inputs:

The new candidate sample (x_c, y_c) and α_i, $i \in S$.

Outputs:

Updated α_i, $i \in S$.

1: Read the new candidate sample (x_c, y_c) to obtain (x_c, y_c'), let $T_{new} = \{(x_c, y_c', +1), (x_c, y_c', -1)\}$ and set the weights of the samples in T_{new} as α_c = 0 and compute $g_i, i \in T_{new}$ according to Equation 8.

2: Update $T \leftarrow T \cup T_{new}$, $g_i \leftarrow \frac{l}{l+1} g_i, i \in T$, $b \leftarrow \frac{l}{l+1} b$ and $\rho \leftarrow \frac{l}{l+1} \rho$.

3: Compute N based on $\hat{N}$.

4: **do** // see RAIA

5: Compute $\gamma^c, \zeta_i^c, i \in R \cup E$ and $\Delta\alpha_c^{\max}$.

6: Update α_i, $i \in S$, g_i, $i \in R \cup E$, the sets S, R and E.

7: Update the inverse matrix N.

8: **while** g_c<0 and α_c<1

9: Compute $\hat{N}$ based on N.

10: **do** // see ARA

11: Compute $\hat{\gamma}$, $\hat{\zeta}_i, i \in R \cup E$ and $\Delta\eta^*$.

12: Update α_i, $i \in S$, g_i, $i \in R \cup E$, the sets S, R and E.

13: Update the inverse matrix $\hat{N}$

14: **while** $\Sigma_{i \in S}\ \alpha_i \neq \nu(l+1)$

15: Ready for the next new candidate sample, go back to step1.

The AIOSVR learning algorithm mainly includes two parts: The first part is Relaxed Adiabatic Incremental Adjustments (RAIA) and the second part is Accurate Restoration Adjustments (ARA). These two parts can avoid the infeasible updating path as far as possible.

RAIA:

During the adiabatic incremental adjustments for α_c, in order to keep all the training samples satisfying the KKT conditions, based on Equation 8 to10, we have:

$$\Delta g_i = \sum_{j \in S} \Delta\alpha_j Q_{ij} + \Delta\alpha_c Q_{ic} + z_i \Delta b + \Delta\rho = 0, \forall i \in S \tag{11}$$

$$\sum_{j \in S} z_j \Delta\alpha_j + z_c \Delta\alpha_c = 0 \tag{12}$$

$$\sum_{j \in S} \Delta\alpha_j + \Delta\alpha_c = \nu \tag{13}$$

Initially, α_c is set as α_c = 0.

If the samples in the set S has the same z_i and z_c of the added new candidate sample is the same as z_i, then Equation 12 is changed into:

$$\sum_{j \in S} \Delta\alpha_j + \Delta\alpha_c = 0 \tag{14}$$

If ν ≠ 0, it is obvious that Equation 13 and 14 cannot hold simultaneously. We call this the contradiction 1, which is shown in Table 1.

The existence of contradictions will cause the fact that α_i cannot be adjusted effectively. To address this issue, we first give up Equation 10 and then a strategy is utilized to restore Equation 10. Giving up Equation 10 means $\Delta\rho$ = 0, therefore, referring to C&P algorithm, this step is called Relaxed Adiabatic Incremental Adjustments (RAIA).

Define $z_s = [z_1, \ldots, z_r]^T$ and $\Delta\alpha_s = [\Delta\alpha_1, \ldots, \Delta\alpha_r]^T$, then Equation 11 and 12 can be further rewritten as the following matrix form:

$$\underbrace{\begin{bmatrix} 0 & z_S^T \\ z_S & Q_{SS} \end{bmatrix}}_{M} \cdot \begin{bmatrix} \Delta b \\ \Delta\alpha_S \end{bmatrix} = -\begin{bmatrix} z_c \\ Q_{Sc} \end{bmatrix} \cdot \Delta\alpha_c \tag{15}$$

Let $N = M^{-1}$ and then we have:

$$\begin{bmatrix} \Delta b \\ \Delta\alpha_S \end{bmatrix} = -N \cdot \begin{bmatrix} z_c \\ Q_{Sc} \end{bmatrix} \cdot \Delta\alpha_c \overset{def}{=} \underbrace{\begin{bmatrix} \gamma_b^c \\ \gamma_S^c \end{bmatrix}}_{\gamma^c} \cdot \Delta\alpha_c \tag{16}$$

where, γ_b^c stands for the dimension corresponding to b in the column vector γ^c and it is similar for γ_S^c.

Finally, substituting Equation 16 into 11, we have:

$$\Delta g_i = \left(\sum_{j \in S_S} \gamma_j^c Q_{ij} + Q_{ic} + z_i \gamma_b^c \right) \cdot \Delta\alpha_c \overset{def}{=} \zeta_i^c \Delta\alpha_c, \forall i \in T \tag{17}$$

Table 1. Two cases of contradictions existed in RAIA

z_c	z_i	Conflict (Yes or No)
+1	+1	Yes
+1	−1	No
−1	+1	No
−1	−1	Yes

It is obvious that $\zeta_i^c = 0, \forall i \in S$.

Note that the RAIA will degenerate into the C&P algorithm when $\Delta\rho = 0$. Therefore, for RAIA, similar to C&P algorithm, we can compute the maximal increment $\Delta\alpha_c^{\max}$ and update α_i, $i \in S$, g_i, $i \in R \cup E$, the sets S, R, E and the inverse matrix N accordingly.

ARA:

After the RAIA, a new strategy is utilized to restore Equation 10 accurately, so this step is called Accurate Restoration Adjustments (ARA).

During the ARA process, $\Sigma_{i \in S}\ \alpha_i$ is gradually adjusted to restore Equation 10, so that all the training samples still satisfying the KKT conditions. The weights of the training samples in the set S, the Lagrange multipliers b and ρ should also be adjusted accordingly. Based on Equation 8 to 10, we have:

$$\Delta g_i = \sum_{j \in S} \Delta\alpha_j Q_{ij} + z_i \Delta b + \Delta\rho = 0, \forall i \in S \tag{18}$$

$$\sum_{j \in S} z_j \Delta\alpha_j = 0 \tag{19}$$

$$\sum_{j \in S} \Delta\alpha_j = -\Delta\eta \tag{20}$$

If the training samples in the set S has the same z_i, it is clear that Equation 19 and 20 cannot hold simultaneously. We call this the contradiction 2, which is shown in Table 2. To avoid this contradiction, Equation 20 is changed into:

$$\sum_{j \in S} \Delta\alpha_j = -(\vartheta\Delta\rho + \Delta\eta) \tag{21}$$

where, $\Delta\eta$ is the introduced new variable to adjust $\Sigma_{i \in S}\alpha_i$, ϑ is any negative number, $\vartheta\Delta\rho$ is an extra term. The purpose of using $\vartheta\Delta\rho+\Delta\eta$ is to prevent the occurrence of contradiction 2.

Further, Equation 18, 19 and 21 can be rewritten as the following matrix form:

$$\underbrace{\begin{bmatrix} 0 & 0 & z_S^T \\ 0 & \vartheta & e_S^T \\ z_S & e_S & Q_{SS} \end{bmatrix}}_{\hat{M}} \cdot \begin{bmatrix} \Delta b \\ \Delta\rho \\ \Delta\alpha_S \end{bmatrix} = -\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \cdot \Delta\eta \tag{22}$$

Table 2. Two cases of contradictions existed in ARA

z_i		Conflict (Yes or No)
+1	+1	Yes
−1	−1	Yes
+1	−1	No
−1	+1	No

Let $\hat{N} = \hat{M}^{-1}$ and then we have:

$$\begin{bmatrix} \Delta b \\ \Delta\rho \\ \Delta\alpha_S \end{bmatrix} = -\hat{N} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \cdot \Delta\eta \overset{def}{=} \underbrace{\begin{bmatrix} \hat{\gamma}_b \\ \hat{\gamma}_\rho \\ \hat{\gamma}_S \end{bmatrix}}_{\hat{\gamma}} \cdot \Delta\eta \tag{23}$$

where, $\hat{\gamma}_b$ stands for the dimension corresponding to b in the column vector γ and it is similar for $\hat{\gamma}_\rho$ and $\hat{\gamma}_S$.

Finally, substituting Equation 23 into 18, we have:

$$\Delta g_i = \left(\sum_{j \in S} \hat{\gamma}_j Q_{ij} + z_i \hat{\gamma}_b + \hat{\gamma}_\rho \right) \cdot \Delta\eta \overset{def}{=} \hat{\zeta}_i \cdot \Delta\eta, \forall i \in T \tag{24}$$

It is obvious that $\hat{\zeta}_i = 0, \forall i \in S$. From Equation 21 and 23, we have $\sum_{j \in S} \Delta\alpha_j = -(\vartheta\hat{\gamma}_\rho + 1)\Delta\eta$, which means the adjustment of $\Sigma_{j \in S}\Delta\alpha_j$ can be achieved by $\Delta\eta$.

Computing the Critical Adjustment Quantity $\Delta\eta^*$:

The critical adjustment quantity $\Delta\eta^*$ is computed to ensure that only a training sample will migrate among the sets S, E and R. This strategy can address the problem when directly apply the ARA to obtain the new optimal solution of Equation 5. If $\Sigma_{i \in S}\ \alpha_i > \nu(l+1)$, compute the maximal adjustments $\Delta\eta^{\max}$ and let $\Delta\eta^* = \Delta\eta^{\max}$; if $\Sigma_{i \in S}\ \alpha_i < \nu(l+1)$, compute the minimal adjustments $\Delta\eta^{\min}$ and let $\Delta\eta^* = \Delta\eta^{\min}$. Three cases should be considered to account for such structural changes.

***Case* 1:** A certain training sample migrates from the set S to the set E or the set R.

When $\Sigma_{i \in S}\ \alpha_i > \nu(l+1)$, the possible weight updates are:

$$\Delta\alpha_i^{\max} = \begin{cases} 1 - \alpha_i, & i \in I_S^+ \\ -\alpha_i, & i \in I_S^- \end{cases}$$

where, $I_S^+ = \{\hat{\gamma}_i > 0, \forall i \in S\}$ and $I_S^- = \{\hat{\gamma}_i < 0, \forall i \in S\}$.

Thus the maximal possible $\Delta\eta^{Case1}$ is:

$$\Delta\eta^{Case1} = \min(\Delta\alpha_i^{\max} / \hat{\gamma}_i, \forall i \in I_S^+ \cup I_S^-)$$

When $\Sigma_{i\in S}\ \alpha_i < \nu(l+1)$, the possible weight updates are:

$$\Delta\alpha_i^{\max} = \begin{cases} -\alpha_i, & i \in I_S^+ \\ 1-\alpha_i, & i \in I_S^- \end{cases}$$

Thus the minimal possible $\Delta\hat{\eta}^{Case1}$ is:

$$\Delta\hat{\eta}^{Case1} = \max(\Delta\alpha_i^{\max} / \hat{\gamma}_i, \forall i \in I_S^+ \cup I_S^-)$$

***Case* 2:** A certain training sample migrates from the set *E* or the set *R* to the set *S*.

When $\Sigma_{i\in S}\ \alpha_i > \nu(l+1)$, the maximal possible $\Delta\eta^{Case2}$ is:

$$\Delta\eta^{Case2} = \min(-g_i / \hat{\zeta}_i, \forall i \in I_E^+ \cup I_R^-)$$

where, $I_E^+ = \{\hat{\zeta}_i > 0, \forall i \in E\}$ and $I_R^- = \{\hat{\zeta}_i < 0, \forall i \in R\}$.

When $\Sigma_{i\in S}\ \alpha_i < \nu(l+1)$, the minimal possible $\Delta\hat{\eta}^{Case2}$ is:

$$\Delta\hat{\eta}^{Case2} = \max(-g_i / \hat{\zeta}_i, \forall i \in I_E^- \cup I_R^+)$$

where, $I_E^- = \{\hat{\zeta}_i < 0, \forall i \in E\}$ and $I_R^+ = \{\hat{\zeta}_i > 0, \forall i \in R\}$.

***Case* 3:** When $\Sigma_{i\in S}\ \alpha_i = \nu(l+1)$, which means the termination condition is met, then the critical adjustment quantity $\Delta\eta^{Case3}$ in case 3 is:

$$\Delta\eta^{Case3} = \frac{\sum_{i\in S}\alpha_i - \nu(l+1)}{\vartheta\hat{\gamma}_\rho + 1}$$

Finally, if $\Sigma_{i\in S}\ \alpha_i > \nu(l+1)$, the smallest value:

$$\Delta\eta^* = \min(\Delta\eta^{Case1}, \Delta\eta^{Case2}, \Delta\eta^{Case3}) \quad (25)$$

will constitute the maximal incremental adjustments of $\Delta\eta$; otherwise, the largest value:

$$\Delta\eta^* = \max(\Delta\hat{\eta}^{Case1}, \Delta\hat{\eta}^{Case2}, \Delta\eta^{Case3}) \quad (26)$$

will constitute the minimal incremental adjustments of $\Delta\eta$.

After the critical adjustment quantity $\Delta\eta^*$ is determined, we can update α_i, $i\in S$, g_i, $i\in R\cup E$, the sets *S*, *R*, *E* similar to C&P algorithm.

Efficiently Updating the Inverse Matrix $\hat{N}$:

Once a training sample is either removed from or added to the set *S*, the inverse matrix $\hat{N}$ should be changed accordingly. Fortunately, based on Lemma 2, we can update the inverse matrix $\hat{N}$ efficiently without solving the inverse $\hat{N}$ directly.

Lemma 2 Suppose a (s+1)× (s+1) matrix B can be partitioned into a block form:

$$B = \begin{bmatrix} A & \eta_t \\ \eta_t^T & Q_{tt} \end{bmatrix}$$

where, A is $s\times s$ matrix and A is invertible, $\eta_t = [Q_{1t},\ldots,Q_{st}]^T$, $Q_{tt} \neq 0$ is a constant.

Then, the inverse matrix of B can be expanded as follows:

$$B^{-1} = \begin{bmatrix} A^{-1} & 0 \\ 0^{\mathrm{T}} & 0 \end{bmatrix} + \frac{1}{k}\begin{bmatrix} \gamma_t \\ 1 \end{bmatrix}\cdot\begin{bmatrix} \gamma_t \\ 1 \end{bmatrix}^{\mathrm{T}}$$

where, $\gamma_t = -A^{-1}\eta_t$ and $k = \eta_t^{\mathrm{T}}\gamma_t + Q_{tt}$.

Furthermore, if B is invertible and $(B^{-1})_{tt} \neq 0$, $t = s$+1, then the inverse of matrix of A can be contracted as follows:

$$A^{-1} = (B^{-1})_{\backslash tt} - \frac{\left((B^{-1})_{*t}\cdot(B^{-1})_{t*}\right)_{\backslash tt}}{(B^{-1})_{tt}}$$

where,* $\neq t$.

It can be easily verified that $BB^{-1} = I_{s+1}$ and $AA^{-1} = I_s$. The detailed proof of Lemma 2 can be found in Laskov *et al*. (2006), it is omitted here.

Based on Lemma 2, if a sample (x_t, y_t', z_t) is removed from the set *S*, then $\hat{N}$ can be contracted as follows:

$$\hat{N} \leftarrow \hat{N}_{\backslash tt} - \frac{(\hat{N}_{*t}\cdot\hat{N}_{t*})_{\backslash tt}}{\hat{N}_{tt}} \quad (27)$$

Similarly, if a sample (x_t, y_t', z_t) is added to the set *S*, then $\hat{N}$ can be expanded as follows:

$$\hat{N} \leftarrow \begin{bmatrix} \hat{N} & 0 \\ 0^T & 0 \end{bmatrix} + \frac{1}{\bar{\zeta}_t^t}\begin{bmatrix} \gamma_t \\ 1 \end{bmatrix}\cdot\begin{bmatrix} \gamma_t \\ 1 \end{bmatrix}^T \quad (28)$$

where, $\bar{\zeta}_t^t = \sum_{j\in S}\bar{\gamma}_j^t Q_{tj} + y_t\bar{\gamma}_b^t + \bar{\gamma}_\rho^t + Q_{tt}$ and

$$\gamma_t = \begin{bmatrix} \bar{\gamma}_b^t \\ \bar{\gamma}_\rho^t \\ \bar{\gamma}_S^t \end{bmatrix} = -\hat{N}\cdot\begin{bmatrix} z_t \\ 1 \\ Q_{St} \end{bmatrix}.$$

Preparations for the Next Round of Adjustments:

From Algorithm 1, it is obvious that we should prepare the inverse matrix $\hat{N}$ for the next round of ARA

after RAIA. Similarly, we should also prepare the inverse matrix N for the next round of RAIA after ARA.

Fortunately, based on Lemma 2, the inverse matrix $\hat{N}$ can be expanded as follows:

$$\hat{N} \leftarrow \begin{bmatrix} N & 0 \\ 0^T & 0 \end{bmatrix} + \frac{1}{k_1}\begin{bmatrix} A \\ 1 \end{bmatrix}\begin{bmatrix} A^T & 1 \end{bmatrix} \tag{29}$$

where, $k_1 = \vartheta - \begin{bmatrix} 0 & e_S \end{bmatrix} \cdot N \cdot \begin{bmatrix} 0 \\ e_S^T \end{bmatrix}$ and $A = -N \cdot \begin{bmatrix} 0 \\ e_S^T \end{bmatrix}$.

Similarly, we can obtain the inverse matrix N from the following three steps.

First, based on Lemma 2, compute the inverse matrix $R \overset{def}{=} Q_{SS}^{-1}$ by using the contracted rules as follows:

$$R \leftarrow \hat{N}_{\setminus\rho\rho} - \frac{1}{\hat{N}_{\rho\rho}}(\hat{N}_{*\rho} \cdot \hat{N}_{\rho *})_{\setminus\rho\rho},\ R \leftarrow R_{\setminus bb} - \frac{1}{R_{bb}}\left(R_{*b} \cdot R_{b*}\right)_{\setminus bb}$$

Second, Update the inverse matrix of Q_{SS} by the rule $R \leftarrow \frac{l+1}{l}R$.

Finally, based on Lemma 2, the inverse matrix N can be expanded as follows:

$$N \leftarrow \begin{bmatrix} R & 0 \\ 0^T & 0 \end{bmatrix} + \frac{1}{k_2}\begin{bmatrix} B \\ 1 \end{bmatrix}\begin{bmatrix} B^T & 1 \end{bmatrix} \tag{30}$$

where, $k_2 = z_S^T \cdot B$ and $B = -R \cdot z_S$.

Similar to Gu and Sheng (2013), we can also prove the feasibility and finite convergence of the AIOSVR learning algorithm. The detailed proof is omitted here. The feasibility of the AIOSVR learning algorithm ensures that there always exist the inverse matrices N and $\hat{N}$ during the adiabatic incremental adjustments and the set S will always be nonempty during the RAIA and ARA. The finite convergence analysis ensures that the AIOSVR learning algorithm will converge to the optimal solution of the minimization problem within finite steps. Therefore, the AIOSVR learning algorithm is effective and reliable.

Glutamic Acid Fed-Batch Fermentation Process

Glutamic acid fed-batch fermentation is a rather complicated nonlinear process. Its corresponding fermentation system is relatively complex (Desai *et al*., 2006). However, with the rapid development of modern technology, glutamic acid fed-batch fermentation is equipped with advanced devices, for example, off-gas analyzer and control cabinet. These devices have greatly enhanced the automation of fermentation process.

Glutamic Acid Fed-batch Fermentation System

Glutamic acid fed-batch fermentation system is mainly composed by fermenter, off-gas analyzer, electronic balance, control cabinet and industrial computer.

The fermenter (Model BIOTECH-5BG-5L) is made in Shanghai Baoxin Biotechnology Equipment Co., Ltd, China. The volume of fermenter is 5L and equipped with temperature sensor, mixing controller, foam electrode, pH electrode and DO electrode. An off-gas analyzer (Model LKM2000A, made in Lokas Co., Ltd, Korea) is connected with the fermenter to measure the CO_2 and O_2 concentration under incomplete pressure state. The main function of the electronic balance is to measure the ammonia consumption and substrate quantity to be added in the glutamic acid fermentation process. The control cabinet is in charge of collecting data, transmitting data and controlling execution counterparts. The industrial computer is used to monitor and store data produced in the glutamic acid fermentation process.

Seeds Culture and Fermentation Condition

The seeds (Corynebacterium Glutamicum S9114) are offered by key laboratory of industrial biotechnology, ministry of education, Jiangnan University. The seeds are added into a shake flask filled with liquid nutrient medium and they are adequately cultivated for 8 to 10 h under the condition of 32°C and 200 r/min. Then, the seeds are added into the fermenter, which has approximately 3.4L liquid fermentation nutrient medium and the pressure of the fermenter is kept in 0.07MPa. The initial pH of the liquid fermentation nutrient medium is in the range of 7.0 to 7.1. In order to keep the pH in the range of 7.1±0.1 during the whole fermentation process, 25% ammonia is automatically added, which also offers necessary ammonia for glutamic acid synthesis. Moreover, 50% glucose is added in the fermentation process to offer necessary glucose concentration for glutamic acid growth. Furthermore, in order to avoid the step change of substrate concentration, a modified feed-rate profile is utilized in this study, which is shown in Fig. 2.

Variables in Glutamic Acid Fed-batch Fermentation Process

There are three different kinds of variable in glutamic acid fed-batch fermentation process: Physical variable, chemical variable and biological variable. The physical variable, such as temperature (°C), pressure (MPa) and air flow (m^3/h), etc., can be measured online. Similarly, the chemical variable, such as pH and Dissolved Oxygen (DO), can also be measured online by certain electrode.

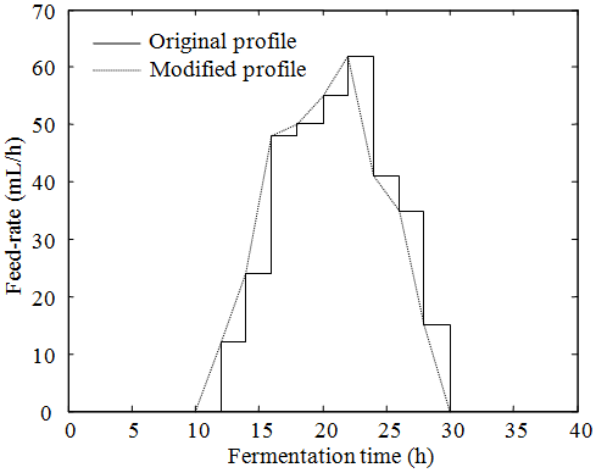

Fig. 2. Feed-rate profiles

However, the biological variable, such as biomass concentration, glutamic acid concentration and substrate concentration, cannot be measured online. Generally, these three biological variables can only be analyzed offline every two hours. Fortunately, with the help of soft sensor modelling, they can be measured and tuned online.

In our experiment, such variables as temperature, pressure, air flow, pH and DO keep constant. The mixing speed of motor can control the DO concentration. Several batch experiments are carried out under the condition of keeping 10%, 20, 30 and 50% DO concentration, respectively.

Soft Sensor Modelling of Biomass Concentration during Fermentation using AIOSVR Learning Algorithm

Data Preprocessing

The data can be divided into offline data and online data. Generally, the temperature, pressure, air flow and pH are almost constant. Therefore, they cannot be treated as the input variable of the soft sensor modelling.

The offline data is measured every two hours, so the same sampling period should be used for online data. In addition, the glucose feed-rate quantity and ammonia consumption use the same way as mentioned above. The sampling period of *OUR* and *CER* is 5 min, so the oxygen uptake O_{2i} and CO_2 production CO_{2i} per 2 h are calculated as follows:

$$O_{2i} = V \sum_{j=24i}^{24(i+1)} OUR_j / 12 \tag{31}$$

$$CO_{2i} = V \sum_{j=24i}^{24(i+1)} CER_j / 12 \tag{32}$$

Table 3. The result of input variables using PCA

Principle component variable	Eigenvalue	Contribution (%)
$X_{i\text{-}1}$	2.87	19.02
$X_{i\text{-}2}$	2.23	14.78
$X_{i\text{-}3}$	0.25	1.66
$S_{i\text{-}1}$	1.92	12.72
$S_{i\text{-}2}$	1.35	8.95
$S_{i\text{-}3}$	0.35	2.32
F_i	1.98	13.12
$F_{i\text{-}1}$	1.58	10.47
$F_{i\text{-}2}$	0.14	0.93
O_i	2.36	15.64
$O_{i\text{-}1}$	0.06	0.40

Selection of Input Variables

The glutamic acid growth mainly depends on biomass concentration and substrate concentration by inner mechanism analysis (Zhang *et al*., 2005). However, different DO leads to different cell activity, which means that the oxygen uptake O_{2i} should be treated as an input variable of the soft sensor modelling. Therefore, there are three different kinds of input variables in the soft sensor modelling of biomass concentration: Biomass concentration X_i, substrate concentration S_i, glucose feed-rate F_i and oxygen uptake O_i.

In our simulation experiment, the following variables $X_{i\text{-}1}$, $X_{i\text{-}2}$, $X_{i\text{-}3}$, $S_{i\text{-}1}$, $S_{i\text{-}2}$, $S_{i\text{-}3}$, F_i, $F_{i\text{-}1}$, $F_{i\text{-}2}$, $F_{i\text{-}3}$ and O_i, $O_{i\text{-}1}$ are selected as the input variables of the soft sensor modelling. The result of input variables using Principle Component Analysis (PCA) (Li *et al*., 2008) is shown in Table 3.

According to the characteristic of fermentation and the contribution of each principle component variable in Table 3, the input variables $X_{i\text{-}3}$, $S_{i\text{-}3}$, $F_{i\text{-}2}$ and $O_{i\text{-}1}$ are ignored. So $X_{i\text{-}1}$, $X_{i\text{-}2}$, $S_{i\text{-}1}$, $S_{i\text{-}2}$, F_i, $F_{i\text{-}1}$ and O_i are selected as the input vector x_i of the expanded training sample set T.

Setting of Model Parameters

In order to ensure the matrix Q_{SS} is always invertible, $K(x_i, x_j) = \exp(-\|x_i\text{-}x_j\|/2\sigma^2)$ is used as the kernel function, where the kernel width parameter is set as σ = 0.7.71. Due to the main function of the parameter C is to transform the output y_i into y_i', for convenience, C is set as 100. In addition, from Equation 23, it is easy to verify $\hat{\gamma}_b$, $\hat{\gamma}_\rho$ and $\hat{\gamma}_i, i \in S$ have the same denominator $\det(\hat{M})$ and ϑ only correlates with $\det(\hat{M})$. Therefore, ϑ can determine $\Delta\eta^*$, but is independent with the structural changes of the sets S, R and E. So the parameter ϑ is fixed at -1. The parameter ν of ν-SVR and AIOSVR is set as 0.3.

Training and Testing of Data

Six batches of glutamic acid fermentation data are selected from the above experiments and each batch of

data can represent the whole fermentation process. The data includes offline analyzed data and online measured data. Five batches are used to train the model and the remaining one batch is used to test the model. The detailed steps of training and testing of data using AIOSVR learning algorithm is listed as follows:

***Step* 1:** Select the input and output of the expanded training sample set T. Here, the input is selected as $x_i = [X_{i\text{-}1}, X_{i\text{-}2}, S_{i\text{-}1}, S_{i\text{-}2}, F_i, F_{i\text{-}1}, O_i]$ and the output is $y_i' = y_i / 100 = X_i$.

***Step* 2:** In order to enhance the speed of operation, the AIOSVR is used to train the model offline based on the five batches of glutamic acid fermentation data.

***Step* 3:** The remaining one batch glutamic acid fermentation data is used to test the model which is already established. Furthermore, the minimum error allowed is set as $\text{Err}_{\text{min}} = 2$.

***Step* 4:** Calculate the prediction error: $e_i = y_i' - \hat{y}_i'$, where y_i' stands for the real value and $\hat{y}_i'$ stands for the predictive value. If $|e_i| < \text{Err}_{\text{min}}$, the model will not be tuned and the data is saved for future use; otherwise, the model will be tuned by training the previously saved data one by one according to Step 2.

***Step* 5:** The next step of prediction is carried out according to Step 4 until the whole fermentation process is over.

Results

All simulation experiments are performed on a 3.1 GHz Inter® Core™ i5-2400 with 4GB RAM and MATLAB 2010a platform.

The prediction results and prediction errors of biomass concentration based on ν-SVR and AIOSVR are shown in Fig. 3 and 4, respectively. Note that biomass concentration refers to OD620 value, which is measured by diluting extracted broth 100 times. The Mean Square Error (MSE) and training time of ν-SVR and AIOSVR are shown in Table 4. Note that T_{24} and T_{32} refer to the training time at fermentation time of 24 and 32 h, respectively.

Discussion

From Fig. 3 and 4, it is clear that a relatively large prediction error occurred at fermentation time of 24 and 32 h, which leads to a larger prediction error. The reason is that the biomass activity becomes stronger at fermentation time of 24 and 32 h. Fortunately, AIOSVR learning algorithm can tune the parameters quickly to cope with this change and obtain smaller prediction error than ν-SVR by online learning. Therefore, the soft sensor modelling based on AIOSVR is of stronger adaptive ability than ν-SVR.

Table 4 demonstrates that the MSE of AIOSVR is around half that of ν-SVR, which means AIOSVR has stronger generalization ability. Furthermore, the training time of AIOSVR is also less than that of ν-SVR.

In summary, the AIOSVR learning algorithm is of better generalization ability and online learning speed than ν-SVR. So it is more suitable for the soft sensor modelling of real fermentation process.

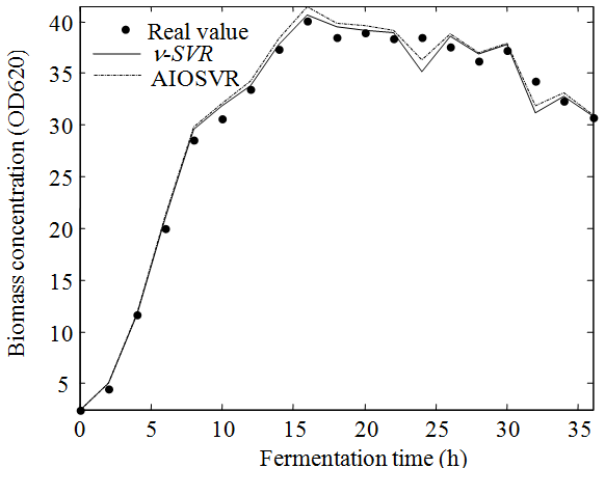

Fig. 3. Prediction results of biomass concentration based on ν-SVR and AIOSVR

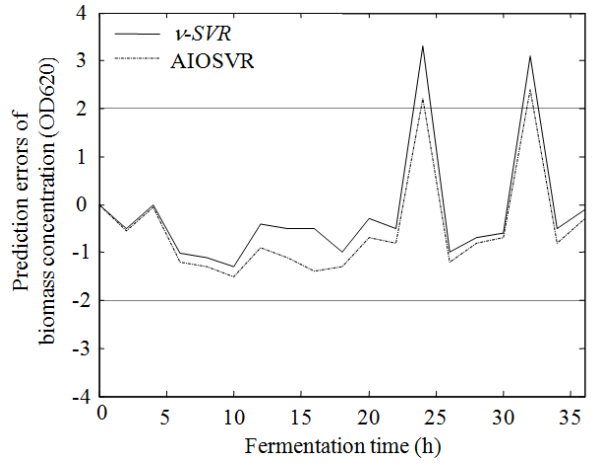

Fig. 4. Prediction errors of biomass concentration based on ν-SVR and AIOSVR

Table 4. MSE and training time of ν-SVR and AIOSVR

	ν-SVR	AIOSVR
MSE	2.616	1.303
T_{24}(s)	48.500	36.300
T_{32}(s)	26.500	18.400

Conclusion

In order to design an online soft sensor modelling for real fermentation process, we first proposed an AIOSVR learning algorithm and then presented its application in soft sensor modelling of biomass concentration for glutamic acid fermentation process.

In theory, the AIOSVR learning algorithm can also be applied in the soft sensor modelling of substrate concentration, product concentration and so on. Therefore, the AIOSVR learning algorithm can be widely used in the soft sensor modelling of microbial fermentation process.

Acknowledgement

The authors will give special thanks to Professor Yuanda Song of Jiangnan University for proofreading the manuscript and giving some constructive comments and suggestions.

Funding Information

This work was supported by the National Natural Science Foundation of China under Grant No.61273131.

Author's Contributions

Binjie Gu: Conceived of the research, designed the research plan, drafted and revised the manuscript.

Feng Pan: Involved in acquisition and analysis of data, manuscript reviewing and provided funding support.

Ethics

This article is original containing unpublished materials. All authors have read and approved the manuscript and no ethical issues involved.

References

Bertsekas, D.P., 2009. Convex Optimization Theory. 1st Edn., Tsinghua University Press, Beijing, ISBN-10: 9787302237600, pp: 167-181.

Cauwenberghs, G. and T. Poggio, 2001. Incremental and decremental support vector machine learning. Advances in Neural Information Processing Systems.

Chang, C.C. and C.J. Lin, 2002. Training ν-support vector regression: Theory and algorithms. Neural Comput., 14: 1959-1977. DOI: 10.1162/089976602760128081

Chen, J. and Y. Li, 2002. The optimization principle and practice of fermentation process. Chemical Industry Press, Beijing.

Desai, K., Y. Badhe, S.S. Tambe and B.D. Kulkarni, 2006. Soft-sensor development for fed-batch bioreactors using support vector regression. Biochem. Eng. J., 27: 225-239. DOI: 10.1016/j.bej.2005.08.002

Feng, R., Y.J. Zhang, Y.Z. Zhang and H.H. Shao, 2004. Drifting modeling method using weighted support vector machines with application to soft sensor. ACTA Automatica Sinica, 30: 436-441.

Gao, X.J., P. Wang, C.Z. Sun, J.Q.Yiand Y.T. Zhang *et al*., 2006. Modeling for penicillin fermentation process based on support vector machine. J. Syst. Simulation, 18: 2052-2055. DOI: 10.3969/j.issn.1004-731X.2006.07.079

Gu, B., J.D. Wang, G.S. Zheng and Y.C. Yu, 2012. Regularization path for ν-support vector classification. IEEE Trans. Neural Netw. Learn. Syst., 23: 800-811. DOI: 10.1109/TNNLS.2012.2183644

Gu, B. and V.S. Sheng, 2013. Feasibility and finite convergence analysis for accurate on-line ν-support vector machine. IEEE Trans. Neural Netw. Learn. Syst., 24: 1304-1315. DOI: 10.1109/TNNLS.2013.2250300

Gu, B., V.S. Sheng, Z.J. Wang, D. Ho and S. Osman *et al*., 2015. Incremental learning for ν-support vector regression. Neural Netw., 67: 140-150. DOI: 10.1016/j.neunet.2015.03.013

Karush, W., 1939. Minima of functions of several variables with inequalities as side conditions. MSc Thesis, University of Chicago, USA.

Laskov, P., C. Gehl, S. Krüger and K.R. Muller, 2006. Incremental support vector learning: Analysis, implementation and applications. J. Machine Learn. Res., 7: 1909-1936.

Li, G.Y., J.P. Zhang, N.X. Wu and J. Liu, 2008. A PCA-based integrative spectrum identification method. J. Northeast Uni. Natural Sci., 29: 1322-1325. DOI: 10.3321/j.issn:1005-3026.2008.09.026

Petrova, M., P. Koprinkova, T. Patarinska and M. Bliznakova, 1998. Neural network modelling of fermentation processes. Bioprocess Eng., 18: 281-287. DOI: 10.1007/s004490050442

Schölkopf, B., A.J. Smola, R.C. Williamson and P.L. Bartlett, 2000. New support vector algorithms. Neural Comput., 12: 1207-1245. DOI: 10.1162/089976600300015565

Shi, Z.P. and F. Pan, 2010. Analysis, control and test technology of fermentation process. Chemical Industry Press, Beijing.

Wang, X.F., Z.Y. Du, J.D. Chen and P. Feng, 2009. Dynamic modeling of biotechnical process based on online support vector machine. J. Comput., 4: 251-258. DOI: 10.4304/jcp.4.3.251-258

Wei, G.H. and X. Yang, 2008. Fermentation Engineering. 1st Edn., Science Press, Beijing, ISBN-10: 9787030210777, pp: 1-5.

Yoshida, F., T. Yamane and K.I. Nakamoto, 1973. Fed-batch hydrocarbon fermentation with colloidal emulsion feed. Biotechnol. Bioeng., 15: 257-270. DOI: 10.1002/bit.260150204

Yu, L.J., 2011. The principle of fermentation engineering and its application. Chemical Industry Press, Beijing.

Zhang, C.Y., P. Gao, Z.P. Shi and Z.G. Mao, 2005. An on-line glutamate production rate for the glutamate fermentation process based on a metabolic reaction model. Food Fermentat. Industries, 31: 54-56. DOI: 10.3321/j.issn:0253-990X.2005.04.014

List of Symbols

R^d denotes the d-dimensional Euclidean space

T denotes matrix transposition

Δ denotes the amount of the change of each variable

Q_{SS} denotes the submatrix of Q with the rows and columns indexed by the set S

Q_{Sc} denotes the subvector of the matrix Q with the rows and columns indexed by the set S and c, respectively

e_s denotes the all ones column vector indexed by the set S

0 denotes the all zeros column vector with proper dimensions

M^{-1} denotes the inverse of the matrix M

'def' above *' = '* denotes the left side of the equal sign is defined as the right side

I_m denotes the identity matrix with m dimensions

$\hat{N}_{it}$ denotes the i_t th row and the i_t th column of the matrix $\hat{N}$, where i_t stands for the corresponding index in $\hat{N}$

$\hat{N}_{\backslash it}$ denotes the submatrix of $\hat{N}$ with deleting the i_t th row and i_t th column, where i_t stands for the corresponding index in $\hat{N}$

X_i denotes the biomass concentration at time i (g/L)

S_i denotes the substrate concentration at time (g/L)

F_i denotes the glucose feed-rate at time i (mL/h)

O_{2i} denotes the oxygen uptake at time i (mol)

CO_{2i} denotes the CO_2 production at time i (mol)

V denotes the volume of broth (m^3)

OUR_i denotes the oxygen utilization rate at time i (mol/m^3/h)

CER_i denotes the CO_2 evolution rate at time i (mol/m^3/h)