Particle Swarm Optimization for the design of $H_\infty$ static output feedbacks

Mohamed Yagoubi*, Guillaume Sandou**

*IRCCyN (Communications and Cybernetic Research Institute of Nantes)
1, rue de la Noé, 44321 Nantes, France
*EMN (Ecole des Mines de Nantes)
4, rue Alfred Kastler, 44307 Nantes, FRANCE
(Tel.: +33 2 51 85 83 27; e-mail: mohamed.yagoubi@mines-nantes.fr)
**SUPELEC Systems Sciences (E3S), Automatic Control Department
3, rue Joliot Curie, 91192 Gif-sur-Yvette, FRANCE
(Tel: +33 1 69 85 13 86; e-mail: guillaume.sandou@supelec.fr)

Abstract: The design of $H_\infty$ reduced order controllers is known to be a non convex optimization problem for which no generic solution exists. In this paper, the use of Particle Swarm Optimization (PSO) for the computation of $H_\infty$ static output feedbacks is investigated. Two approaches are tested. In a first part, a probabilistic-type PSO algorithm is defined for the computation of discrete sets of stabilizing static output feedback controllers. This method relies on a technique for random sample generation in a given domain. It is therefore used for computing a suboptimal $H_\infty$ static output feedback solution. In a second part, the initial optimization problem is solved by PSO, the decision variables being the feedback gains. Results are compared with standard reduced order problem solvers using the COMPlib benchmark examples and appear to be much than satisfactory, proving the great potential of PSO techniques.

Keywords: Reduced order controllers, Particle Swarm Optimization, static output feedback, $H_\infty$ synthesis

1. INTRODUCTION

$H_\infty$ synthesis is an efficient tool in automatic control to compute controllers in a closed-loop framework, achieving high and various performances (Gahinet and Apkarian, 1994, Zhou, et al., 1996). The main drawback of such approaches is the controller order: $H_\infty$ synthesis provides a controller whose order is the same as the synthesis model. A classical way to derive low order controllers is to use a full-order synthesis and to reduce the obtained controller. However, this kind of approach may lead to a high $H_\infty$ norm of the closed-loop system and a high sensitivity to high frequency noises.

Another approach is to solve an $H_\infty$ optimization problem, adding some order constraints on the solution. However, this kind of constraints is expressed by matrices rank constraints. Thus, the reduced-order synthesis problem appears to be a non-convex, non-smooth optimization problem. Several approaches exist, trying to overcome this difficulty:

- Convex approximations of non-convex stability regions, but often obtained from conservative conditions;
- Global nonlinear optimization techniques such as an exhaustive search, which are very expensive;
- Local optimization techniques such as BMI solvers. In that case, the solution strongly depends on the initial controller.

More recently, some new techniques have begun to emerge, adding some random process in the deterministic search algorithm (Arzelier, et al., 2010), and achieving results which are almost similar to those obtained with the HIFOO standard (Burke, et al., 2006). Following this new trend, a new approach is proposed in this paper, based on Particle Swarm Optimization (PSO) to solve the $H_\infty$ static output feedback synthesis problem. PSO was firstly introduced by Eberhart and Kennedy (Eberhart and Kennedy, 1995). This optimization method belongs to the class of approximated stochastic method called metaheuristics methods. With such methods, the optimality of the computed solution can never be guaranteed, but there are strong advantages:

- The structure of costs and constraints is not an essential point as no gradient has to be computed;
- The random part allows escaping from local minima.

Two approaches based on PSO are presented in the sequel. In the first one, a probabilistic-type PSO algorithm, to deal with the problem of building sets of stabilizing static output feedback controllers, is proposed. A suboptimal $H_\infty$ static output feedback solution is then computed obviously by selecting the element achieving the best $H_\infty$ performance in the constructed set. In this approach, the PSO algorithm is proposed to maximize an empirical estimate of the existence probability of static output feedback controllers in some bounded balls moving in the search space. The probability estimate is evaluated based on a random sample generation technique borrowed from (Calafiore, et al., 1999, Calafiore, et al., 2006). This technique produces a finite number of real random matrices uniformly distributed in a given domain.

The second approach deals with the direct optimization of output feedback gains, the optimization variables being the feedback gains.
The paper is organized as follows. The PSO algorithm is presented in section 2. For MIMO systems, a static output feedback may lead to numerous optimization variables and so a perturbed PSO algorithm is also presented. In section 3, a probabilistic-type PSO-based approach leading to discrete sets of stabilizing static output feedback controllers is presented. A direct $H_{\infty}$ optimization of feedback gains is fully described in section 4. In the same section, an indirect application of PSO, based on the method described in section 3, is also proposed for solving the optimal $H_{\infty}$ static output feedback problem. Comparisons with standard algorithms show much than satisfactory results which can be achieved with low computation loads. Finally conclusions and forthcoming works are drawn in section 5.

2. PARTICLE SWARM OPTIMIZATION

2.1 Classical algorithm

PSO was firstly introduced by Eberhart and Kennedy (Eberhart and Kennedy, 1995). This optimization method is inspired by the social behavior of bird flocking or fish schooling. Consider the following optimization problem:

$$\min_{x \in \chi} f(x)$$

$P$ particles are moving in the search space $\chi$. Each of them has its own velocity, and is able to remember where it has found its best performance. Each particle has also some “friends”. The following notations will be used:

- $x_p^k$: position of particle $p$ at iteration $k$;
- $v_p^k$: velocity of particle $p$ at iteration $k$;
- $b_p^k = \arg \min_{x \in \chi} f(x)$: best position found by particle $p$ until iteration $k$;
- $V(x_p^k) \subset \{1, 2, \ldots, P\}$ set of “friend particles” of particle $p$ at iteration $k$;
- $g_p^k = \arg \min_{x \in (b_p^k, \ldots, v_p^k)} f(x)$: best position found by the friend particles of particle $p$ until iteration $k$.

The particles move in the search space $\chi$ according to the following transition rule:

$$v_{p}^{k+1} = c_0 v_p^k + c_1 (b_p^k - x_p^k) + c_2 (g_p^k - x_p^k)$$
$$x_{p}^{k+1} = x_p^k + v_{p}^{k+1}$$

(2)

In this equation:

- $\otimes$ is the element wise multiplication of vectors;
- $c_0$ is the inertia factor;
- $c_1$ (resp. $c_2$) is a random number in the range $[0, \bar{c}_1]$ (resp. $[0, \bar{c}_2]$).

2.2 Perturbed PSO algorithm

The choice of parameters is very important to ensure the satisfying convergence of the algorithm. Lots of work have been done on the topic; see for instance (Shi and Eberhart, 1998; Trellea 2003). However the PSO algorithm may suffer from premature convergence to local minima, especially for optimization problems with numerous variables. This can be the case for static output feedbacks, as a MIMO plant with $r$ measured outputs and $m$ control inputs leads to $r \times m$ optimization variables. To overcome this difficulty, some perturbed versions have been defined. The idea is to add a random movement to the best particle to avoid that behaviour. Following the algorithm given in (van den Bergh and Engelbrecht, 2002), the transition rule (2) for the best particle is reformulated into:

$$v_{p}^{k+1} = c_0 v_p^k + (g_p^k - x_p^k) + \rho^k (1 - 2r_{[0,1]})$$
$$x_{p}^{k+1} = x_p^k + v_{p}^{k+1}$$

(3)

where $r_{[0,1]}$ is a random vector in the range $[0,1]$. The value of $\rho^k$ is updated after each time step by:

$$\rho^{k+1} = \begin{cases} 2 \rho^k & \text{if } nb \_ success > s_c \\ 0.5 \rho^k & \text{if } nb \_ failure > f_c \\ \rho^k & \text{otherwise} \end{cases}$$

(4)

where $nb \_ success$ is the number of consecutive successes and $nb \_ failure$ the number of consecutive failures (a step is a success if the best value found by the particles is enhanced, and a failure otherwise). $s_c$ and $f_c$ are 2 tuning parameters. For more details, see for instance (Van den Bergh and Engelbrecht, 2002; Xinchao, 2010) and references therein.

3. PROBABILISTIC-TYPE PSO FOR STATIC OUTPUT FEEDBACK STABILIZATION

Some attempts to use PSO for the design of automatic control laws have already been presented for instance in (Sandou, 2009). The originality, however, of the method proposed in this paper is twofold:

- It allows building discrete sets of stabilizing static output feedbacks: the use of random sample generation technique makes it possible to better scrutinize the set of stabilizing static output feedbacks which may be nonconvex and consisting of many disjoint domains.
- It takes advantage of probabilistic information (probability estimate of the existence of a static output feedback in a predefined domain) which is not the case of a standard PSO method.

3.1 Problem statement

Consider the state space representation of the plant $\Sigma$: 

$$\begin{align*}

\dot{x}(t) &= A x(t) + B u(t) + D w(t) \\
y(t) &= C x(t) + D w(t)
\end{align*}$$

where $A$, $B$, $C$, and $D$ are matrices of appropriate dimensions. The goal is to design a stabilizing static output feedback $K$ such that the closed-loop system is stable.
\[
\begin{align*}
\dot{x}(t) &= A x(t) + B w(t) + B u(t) \\
\Sigma : z(t) &= C x(t) + D_1 w(t) + D_2 u(t), \\
y(t) &= C x(t) + D_1 w(t),
\end{align*}
\] (5)

where \( x \in \mathbb{R}^n \) is the state vector, \( u \in \mathbb{R}^m \) is the control vector, \( y \in \mathbb{R}^p \) is the measured outputs vector, \( w \in \mathbb{R}^w \) is the input vector (reference and disturbance inputs), and \( z \in \mathbb{R}^r \) is the controlled vector. All matrices are assumed to be of appropriate dimensions. The model (5) is assumed to be stabilizable by static output feedback, that is to say that there exists a control law \( u(t) = K y(t) \) such that the closed-loop of figure 1 is stable.

Fig. 1: Computation of static output feedbacks

The set of stabilizing static output feedback matrices is:

\[
\Gamma = \{ K \in \mathbb{R}^{m \times n} / \Lambda(A + BK C) \subseteq \mathbb{C}^- \} \quad \text{(6)}
\]

where \( \Lambda(M) \) denotes the spectrum (all eigenvalues) of the matrix \( M \). The first problem tackled with in this paper is to build a non-trivial subset \( \Gamma_{\alpha} \subset \Gamma \) of \( n_s \geq 1 \) instances.

### 3.2 Sample generation problem

We describe here the method used for generating real matrix samples \( K \in \mathbb{R}^{m \times n} \) bounded in the \( l_2 \) norm ball of radius \( \delta \) given by:

\[
\Theta_s(\delta, \mathbb{R}^{m \times n}) = \left\{ K \in \mathbb{R}^{m \times n} / \| K \| \leq \delta \right\}, \quad g = 1, 2, \infty \quad \text{(7)}
\]

The problem can be reduced to multiple random vector generation for which the technique borrowed from (Calafiore, et al., 1999) can be used. It is an algorithm that returns a real random vector \( q \in \mathbb{R}^{m} \) uniformly distributed in \( \Theta_s(\delta, \mathbb{R}^{m}) \).

The algorithm is based on a generalized gamma density function \( \tilde{G}_{a,b}(f) \), defined as:

\[
\tilde{G}_{a,b}(f) = \frac{b}{G(a)} f^{a-1} e^{-bf}, \quad f \in \mathbb{R}^+
\] (8)

where \( a \) and \( b \) are given parameters and \( G(a) \) is the gamma function. We report here the algorithm used for this purpose in the Randomized Algorithms Control Toolbox (RATC) (http://ract.sourceforge.net).

Algorithm 1: (Uniform generation in a real \( l_2 \) norm ball)

- Generate \( m \) independent real random scalars \( \xi_i \in \tilde{G}_{1/\xi_s} \);
- Construct the vector \( d \in \mathbb{R}^m \) of components \( d_i = s_i \xi_i \), where \( s_i \) are independent random signs;
- Generate \( h = \varphi^{1/m} \), where \( \varphi \) is uniform in \([0,1]\);
- Return \( q = \delta h d \| d \|_2 \).

Remark 1: The solution for the case \( \Theta(\delta, \mathbb{R}^{m \times n}) \) can also be dealt with using RATC. A singular value decomposition is needed in that case. See (Tempo, et al. 2005) for more details.

### 3.3 Probabilistic-type PSO for static output feedback stabilization

The algorithm proposed in this section uses complementary advantages of:

- The sample generation method described in section 3.2.
- The probabilistic information (which is often neglected in a deterministic context) that consists in an empirical probability estimate of the existence of a static output feedback in some predetermined subsets.
- The PSO (the meta-heuristic method described in section 2) property of escaping from local minima.

The main idea for building a set of stabilizing static output feedback controllers for the system given in (5) is summarized hereafter.

The following lines describe the proposed method:

-For an iteration \( j \), the method starts from an initialization point \( K' \in \Gamma \) that can be computed using a direct application of PSO to optimize a criterion of the form: \( \max \{ \text{Re}(\Lambda(A + BK C)) \} \).

-Define then a search domain bounded in a \( l_2 \) norm ball of radius \( \delta_{K'} \) centred on \( K' \) and given by:

\[
\Theta_{s}^{K'}(\delta_{K'}, \mathbb{R}^{m \times n}) = \left\{ K \in \mathbb{R}^{m \times n} / \| K - K' \|_2 \leq \delta_{K'} \right\}, \quad g = 1, 2, \infty
\] (9)

-A PSO particle \( p \) represents a couple \( (\delta_p, K_p') \) that is a \( l_2 \) norm ball \( \delta_p^{K'}(\delta_p, \mathbb{R}^{m \times n}) \) of radius \( \delta_p \) and centred on \( K_p' \) moving in the search domain \( \Theta_{s}^{K'} \) defined in (9).

-At each iteration \( k \) of the PSO algorithm, a fixed number \( N \) of samples \( \{K_p'\}_{i=1}^{N} \) uniformly distributed in \( \delta_{K'}^{K'} \) is generated using the method described in section 3.2. The PSO algorithm is used to optimise the criterion:
including the radius \( j^i \) and an empirical probability estimate \( \hat{P}_N \) defined by:
\[
\hat{P}_N = \frac{1}{N} \sum_{i=1}^{N} E(K(j))
\]
where \( E(K(j)) = 1 \) when \( K(j) \in \Gamma \) and \( E(K(j)) = 0 \) otherwise.

Finally, the solutions \( K(j) \in \Gamma \) are added to the collection set \( \Gamma_{ns} \) of \( ns \) instances.”

The above algorithm is run several times \( j = 1, \ldots, pso \) with different initialization points \( K(j) \in \Gamma \) and the resulting set of stabilizing controllers is given by:
\[
\Gamma_{ns} = \bigcup_{j=1, \ldots, pso} \Gamma_{ns}^j
\]

3.4 Some results from Complib library

For sake of comparison with existing methods (e.g. Arzelier, et al., 2010), the proposed method is tested on all examples of the database ComPlib (Leibfritz, 2004). It is a library composed of different Linear Time Invariant models and ranging from purely academic problems to more realistic industrial examples. Among them are models of Aircraft (AC), Helicopter (HE), Jet engine (JE), Reactor (REA), Decentralized interconnected systems (DIS), academic tests problems (NN), a Wind energy conversion (WEC), a Binary distillation towers (BDT), a Strings (CSE), a Piezoelectric bimorph actuator (PAS), a Tuned Mass damper (TMD), a Flexible Satellite (FS) and 2D heat Flow (HF2D). This relevant library is now extensively used in order to evaluate the effectiveness of some control design strategies (see for instance (Yagoubi, et al., 2005)).

Figures 2 and 3 show the population of stabilizing gains for the two-parameters examples AC7 and NN5.

![Fig. 2: Set of stabilizing static output feedbacks generated with the probabilistic-type PSO method for example AC7](image)

![Fig. 3: Set of stabilizing static output feedbacks generated with the probabilistic-type PSO method for example NN5](image)

These results were found with \( pso = 8 \) iterations executed from 8 different initializing points \( K(j) \in \Gamma \) computed with a direct application of PSO algorithm and \( (N = 10, \delta = 30, g = 1) \). The values of the PSO algorithm parameters are given in section 4.1.

Similar to the algorithms proposed in (Arzelier, et al., 2010), the proposed method allows to detect the exact shape of the set of stabilizing static output feedback controllers.

Remark 2: Note that algorithm 2 has a low computation load, as it only evaluates the stability of systems. The number of stabilizing static output feedbacks found depends, however, on the initialization points, the iterations number and the choice of the parameters \( (N, \delta, g) \).

4. OPTIMIZATION OF STATIC H\(_\infty\) OUTPUT FEEDBACKS

4.1 Direct PSO method

In this section, the direct optimization of static H\(_\infty\) output feedbacks for a given MIMO plant by a perturbed PSO algorithm is tested. Consider once again the state space representation given by (5). The problem refers to the following optimization problem:
\[
\min_{K \in \mathbb{R}^{mxn}} \|T_{w \rightarrow z}\|_{\infty}, \text{s.t. } u(t) = K y(t),
\]
where \( \|T_{w \rightarrow z}\|_{\infty} \) denotes the H\(_\infty\) norm of the transfer from \( w \) to \( z \). Of course, the closed-loop may be unstable for some values of matrix \( K \in \mathbb{R}^{mxn} \). To tackle this difficulty, the optimization problem is reformulated into:
\[
\min_{K \in \mathbb{R}^{mxn}} J(K),
\]
where:
\[
J(K) = -1/ \|T_{w \rightarrow z}\|_{\infty},
\]
if the closed-loop of figure 1 is stable, and:

12584
\[ J(K) = \max(\Re(\lambda(T_{\text{w-x}}(s)))) \]  
otherwise. \( \lambda(T_{\text{w-x}}(s)) \) denotes the set of the poles of the closed-loop transfer \( T_{\text{w-x}}(s) \). This kind of criterion can be optimized by PSO which does not require any particular formulation of the cost function.

The optimization problem is solved using a perturbed PSO algorithm. For comparison, the algorithm is tested on the benchmark examples given in the COMPlib library. Results obtained with the PSO algorithm have been compared with those obtained with the HIFOO package (Burke, et al. 2006), considered as one of the best effective tool for the synthesis of static output feedback, and those obtained in (Arzelier, et al., 2010). Following these works, the algorithm is not tested for systems that are already open-loop asymptotically stable. Corresponding results are given in table I. In (Arzelier, et al., 2010), several algorithms were tested. The best value achieved so far is kept for the comparison.

For the PSO algorithm, the following values have been chosen for the parameters: \( c_0 = 0.7, \quad \tau_1 = \tau_2 = 1.5, \quad s_0 = 15; f_s = 5 \); Swarm size = 30; Number of iterations: 1000 and initial value of \( \rho : 1 \).

The direct PSO method has been able to stabilize all systems, except A10 (but the other algorithms are not able to do so either). Except examples AC18 and HF2D1...HF217, the PSO algorithm achieves very satisfactory results, with H\(_{\infty}\) norm values very close to the HIFOO solver. For examples AC1, HE1, DIS2, DIS5, REA2, NN2, NN5, NN6, NN7, NN15 and FS the PSO algorithm gives even the best results. Note also that the PSO algorithm has a low computation load, as it only evaluates the H\(_{\infty}\) norm of systems. Further, results could be enhanced again with a fine tuning of the parameters. In the presented tests, the same values have been chosen for each example, whereas they should be adapted to each case.

### 4.2 Indirect PSO method

A suboptimal H\(_{\infty}\) static output feedback solution can also be computed thanks to Algorithm 2. In fact, this is possible by selecting, from the constructed set of stabilizing static output feedbacks \( \Gamma_{\omega} \), the element achieving the best H\(_{\infty}\) performance. Numerical results, obtained using the same parameters \( (N_\text{w}, \delta_\text{w}, \delta) \) as in section 3.4 for the COMPlib library examples are given in Table I (see Appendix).

### 5. CONCLUSIONS AND FORTHCOMING WORKS

In this paper, two design methods of H\(_{\infty}\) reduced order controllers by using PSO algorithm have been proposed. First, a new method based on a probabilistic-type PSO algorithm, that computes discrete set of stabilizing static output feedback controllers, has been given. A suboptimal H\(_{\infty}\) static output feedback solution is computed obviously by selecting the element achieving the best H\(_{\infty}\) performance in the constructed set. Second, the initial optimization problem has been solved by a direct PSO method, the decision variables being the feedback gains.

Moreover, the numerical results presented in this paper show comparisons with standard reduced order problem solvers using the COMPlib benchmark examples. The proposed methods seem promising when compared with HIFOO and the mixed Randomized/LMI algorithm proposed in (Arzelier, et al., 2010).

Therefore, a forthcoming work will deal with similar methods, incorporating probabilistic strategies into PSO algorithms, to construct new algorithms that can outperform existing methods in designing multi-objective structured output feedback controllers.

### APPENDIX

**Table I. Computation of H\(_{\infty}\) static output feedbacks**

<table>
<thead>
<tr>
<th>Example</th>
<th>n</th>
<th>m</th>
<th>r</th>
<th>HIFOO</th>
<th>Arzelier, et al.</th>
<th>Direct PSO</th>
<th>Indirect PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC1</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>4.14 \times 10^1</td>
<td>1.76 \times 10^9</td>
<td>4.7 \times 10^{-1}</td>
<td>1.2 \times 10^{-1}</td>
</tr>
<tr>
<td>AC2</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>0.1115</td>
<td>0.1115</td>
<td>0.1115</td>
<td>0.112</td>
</tr>
<tr>
<td>AC5</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>669.56</td>
<td>661.7</td>
<td>665.09</td>
<td>684.32</td>
</tr>
<tr>
<td>AC9</td>
<td>10</td>
<td>5</td>
<td>4</td>
<td>1.0029</td>
<td>1.0061</td>
<td>1.098</td>
<td>1.12</td>
</tr>
<tr>
<td>AC10</td>
<td>55</td>
<td>2</td>
<td>2</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>AC11</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>2.8335</td>
<td>2.8375</td>
<td>2.8609</td>
<td>2.923</td>
</tr>
<tr>
<td>AC12</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>0.3120</td>
<td>0.6165</td>
<td>0.3134</td>
<td>0.640</td>
</tr>
<tr>
<td>AC13</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>163.33</td>
<td>395.0404</td>
<td>167.36</td>
<td>173.35</td>
</tr>
<tr>
<td>AC17</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>101.7203</td>
<td>319.31</td>
<td>101.96</td>
<td>111.385</td>
</tr>
<tr>
<td>AC18</td>
<td>10</td>
<td>2</td>
<td>2</td>
<td>12.6282</td>
<td>10.6214</td>
<td>27.18</td>
<td>10.639</td>
</tr>
<tr>
<td>HE1</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0.1539</td>
<td>0.1538</td>
<td>0.1529</td>
<td>0.154</td>
</tr>
<tr>
<td>HE3</td>
<td>8</td>
<td>4</td>
<td>6</td>
<td>0.8061</td>
<td>0.8291</td>
<td>0.8399</td>
<td>0.853</td>
</tr>
<tr>
<td>HE4</td>
<td>8</td>
<td>4</td>
<td>6</td>
<td>22.8282</td>
<td>22.8282</td>
<td>23.43</td>
<td>23.512</td>
</tr>
<tr>
<td>HE5</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>8.8952</td>
<td>17.6061</td>
<td>10.0031</td>
<td>8.931</td>
</tr>
<tr>
<td>HE6</td>
<td>20</td>
<td>4</td>
<td>6</td>
<td>192.3445</td>
<td>401.7698</td>
<td>195.86</td>
<td>198.325</td>
</tr>
<tr>
<td>HE7</td>
<td>20</td>
<td>4</td>
<td>6</td>
<td>192.3885</td>
<td>353.9425</td>
<td>194.24</td>
<td>210.931</td>
</tr>
<tr>
<td>DIS2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1.0412</td>
<td>1.0244</td>
<td>1.0255</td>
<td>1.026</td>
</tr>
<tr>
<td>DIS4</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>0.7394</td>
<td>0.7404</td>
<td>0.7863</td>
<td>0.811</td>
</tr>
</tbody>
</table>
REFERENCES


