A Tunable Radial Basis Function Model for Nonlinear System Identification Using Particle Swarm Optimisation

S. Chen[1], X. Hong[2], B.L. Luk[3], C.J. Harris[1]

[1]School of Electronics and Computer Science
University of Southampton, Southampton SO17 1BJ, UK

[2]School of Systems Engineering
University of Reading, Reading RG6 6AY, UK

[3]Department of Manufacturing Engineering and Engineering Management
City University of Hong Kong, Hong Kong, China

Acknowledge: Supports of UK Royal Society and Royal Academy of Engineering

Outline

Outline

Existing Approaches

- Nonlinear optimisation approach: optimise all parameters of nonlinear model together
  - Very “sparse” (small size), but all problems associated with nonlinear optimisation
- Linear optimisation approach: adopt fixed bases and seek a “linear” subset model
  - Orthogonal least squares forward selection: sparse and efficient construction; need to specify RBF variance (via cross validation)
  - Sparse kernel modelling methods: not as sparse as OLS; need to specify kernel variance and other hyperparameters (via cross validation)

Early Orthogonal Least Squares

- Orthogonal least squares methods and their application to non-linear system identification - S. Chen, S. A. Billings and W. Luo - International Journal of Control, 1989
  Google scholar citations: 494 **ISI** citations: 369 (September 2009)

- Orthogonal least squares learning algorithm for radial basis function networks - S. Chen, C. F. N. Cowan and P. M. Grant - IEEE Transactions on Neural Networks, 1991
  Google scholar citations: 1747 **ISI** citations: 1174 (September 2009)

Previous State-of-the-Art

- Optimal experimental design assisted orthogonal least squares
  - S. Chen, X. Hong and C.J. Harris, "Sparse kernel regression modelling using combined locally regularized orthogonal least squares and D-optimality experimental design," *IEEE Trans. Automatic Control*, Vol.48, No.6, pp.1029–1036, 2003
- Local regularisation assisted orthogonal least squares based on leave-one-out mean square error (LROLS-LOO)
  - S. Chen, X. Hong, C.J. Harris and P.M. Sharkey, "Sparse modelling using orthogonal forward regression with PRESS statistic and regularization," *IEEE Trans. Systems, Man and Cybernetics, Part B*, Vol.34, No.2, pp.898–911, 2004

Outline

Combined Linear/Nonlinear Optimisation

- Retain advantage of linear optimisation $\rightarrow$ Use orthogonal forward regression to add bases one by one
- Have tunable bases for enhanced modelling capability $\rightarrow$ Use nonlinear optimisation
- Each stage of OFR, optimise one tunable base, i.e. determine base's nonlinear parameters
- How efficient this combined model construction approach?
  - **Particle swarm optimisation aided OFR for tunable-node RBF models produces smaller model, better generalisation and more efficient model construction over the state-of-the-art LROLS-LOO for constructing fixed-node RBF models**

Outline

NARX System

- We consider NARX system

$$y_k = f_s(y_{k-1}, \cdots, y_{k-m_y}, u_{k-1}, \cdots, u_{k-m_u}) + e_k = f_s(\mathbf{x}_k) + e_k$$

- u_k and y_k: system input and output variables; m_u and m_y: known lags for u_k and y_k; e_k: zero-mean uncorrelated noise; $f_s(\bullet)$: unknown system mapping, and system input vector of known dimension $m = m_y + m_u$:

$$\mathbf{x}_k = [x_{1,k}\ x_{2,k} \cdots x_{m,k}]^T = [y_{k-1} \cdots y_{k-m_y}\ u_{k-1} \cdots u_{k-m_u}]^T$$

- The technique can be extended to the NARMAX system

Outline

Tunable RBF Modelling

- Given training set $D_K = \{\mathbf{x}_k, y_k\}_{k=1}^{K}$, construct M-node RBF model

$$\hat{y}_k^{(M)} = \sum_{i=1}^{M} \theta_i p_i(\mathbf{x}_k) = \mathbf{p}_M^T(k)\boldsymbol{\theta}_M$$

- θ_i are linear weights, and generic RBF node

$$p_i(\mathbf{x}) = \varphi\left(\sqrt{(\mathbf{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i)}\right)$$

- $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i = \text{diag}\{\sigma_{i,1}^2, \cdots, \sigma_{i,m}\}$ are ith centre vector and diagonal covariance matrix; $\varphi(\bullet)$ is chosen basis function

- Regression model on training set D_K

$$\mathbf{y} = \mathbf{P}_M \boldsymbol{\theta}_M + \boldsymbol{\varepsilon}^{(M)}$$

Orthogonal Decomposition

- Orthogonal decomposition of regression matrix: $\mathbf{P}_M = \mathbf{W}_M \mathbf{A}_M$ with

$$\mathbf{A}_M = \begin{bmatrix} 1 & \alpha_{1,2} & \cdots & \alpha_{1,M} \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \alpha_{M-1,M} \\ 0 & \cdots & 0 & 1 \end{bmatrix}$$

- $\mathbf{W}_M = [\mathbf{w}_1 \cdots \mathbf{w}_M]$ is orthogonal, $\mathbf{A}_M \boldsymbol{\theta}_M = \mathbf{g}_M$ and equivalent model

$$\mathbf{y} = \mathbf{W}_M \mathbf{g}_M + \varepsilon^{(M)}$$

- After nth stage of OFR, n bases $\mathbf{W}_n = [\mathbf{w}_1 \cdots \mathbf{w}_n]$ are constructed with related $\mathbf{A}_n$
- Denote kth row of $\mathbf{W}_n$ as $[w_1(k) \cdots w_n(k)]$

Leave-One-Out Cross Validation

- Leave-one-out error

$$\varepsilon_k^{(n,-k)} = \varepsilon_k^{(n)} / \eta_k^{(n)}$$

- Modelling error of *n*-term model

$$\varepsilon_k^{(n)} = \varepsilon_k^{(n-1)} - g_n w_n(k)$$

- Leave-one-out error weighting

$$\eta_k^{(n)} = \eta_k^{(n-1)} - w_n^2(k) / \left(\mathbf{w}_n^T \mathbf{w}_n + \lambda\right)$$

  λ being a regularisation parameter

- A generalisation measure is LOO mean square error

$$J_n = \frac{1}{K} \sum_{k=1}^{K} \left(\varepsilon_k^{(n,-k)}\right)^2$$

Nonlinear Optimisation in OFR

- At nth stage of OFR, determine nth RBF node by solving nonlinear optimisation

$$(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) = \arg \min_{\boldsymbol{\mu}, \boldsymbol{\Sigma}} J_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- There exists an "optimal" model size M such that, for $n \leq M$ J_n decreases as model size n increases while

$$J_M \leq J_{M+1}$$

- Thus OFR construction process is automatically terminated, yielding an M-node RBF model
- We use particle swarm optimisation: a population based stochastic optimisation method (Swarm Intelligence)
  - inspired by social behaviour of bird flocks or fish schools

Outline

Particle Swarm Optimisation

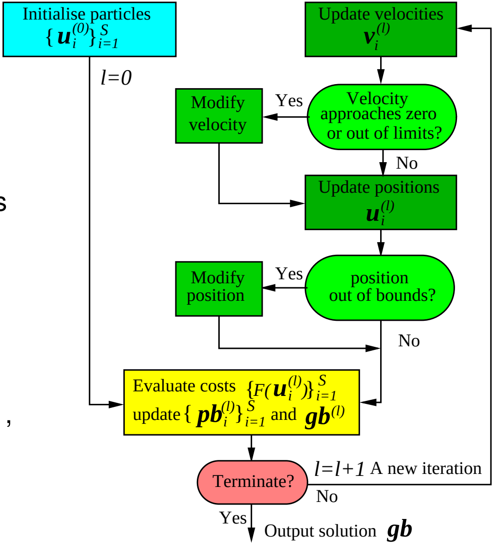
- Solving generic optimisation

$$\mathbf{u}_{\text{opt}} = \arg \min_{\mathbf{u} \in \prod_{j=1}^{m'} \mathsf{P}_j} F(\mathbf{u})$$

$\mathbf{u} = [u_1 \cdots u_{m'}]^T$ is parameter vector to be optimised, $F(\bullet)$ is cost, and search space

$$\prod_{j=1}^{m'} \mathsf{U}_j = \prod_{j=1}^{m'} [U_{j,\min},\ U_{j,\max}]$$

- A swarm of particles, $\{\mathbf{u}_i^{(l)}\}_{i=1}^{S}$, are evolved in search space, where S is swarm size and l denotes iteration index



PSO Algorithm Adopted

- Each particle remembers its best position visited – *cognitive information*, $\mathbf{pb}_i^{(l)}$, $1 \le i \le S$
- Every particle knows best position visited among entire swarm – *social information*, $\mathbf{gb}^{(l)}$
- Each particle has a velocity $\mathbf{v}_i^{(l)}$ to direct its "flying", and

$$\mathbf{v}_i^{(l)} \in \prod_{j=1}^{m'} \mathrm{V}_j = \prod_{j=1}^{m'} [-V_{j,\max},\ V_{j,\max}]$$

- In our application, $m' = 2m$, each $\mathbf{u}_i^{(l)}$ contains a candidate solution for $(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$, and cost function $F(\mathbf{u}) = J_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

Outline

PSO Aided Tunable RBF Construction

- *a) Swarm initialisation*: Set iteration index $l = 0$ and randomly generate $\{\mathbf{u}_i^{(l)}\}_{i=1}^{S}$ in search space $\prod_{j=1}^{m'} \mathrm{U}_j$;
- *b) Swarm evaluation*: Particle $\mathbf{u}_i^{(l)}$ has cost $F(\mathbf{u}_i^{(l)})$, based on which $\mathbf{pb}_i^{(l)}$, $1 \leq i \leq S$, and $\mathbf{gb}^{(l)}$ are updated
- *c) Swarm update*: Velocities and positions are updated

$$\mathbf{v}_i^{(l+1)} = w_{\mathrm{I}} * \mathbf{v}_i^{(l)} + rand() * c_1 * (\mathbf{pb}_i^{(l)} - \mathbf{u}_i^{(l)}) + rand() * c_2 * (\mathbf{gb}^{(l)} - \mathbf{u}_i^{(l)})$$

$$\mathbf{u}_i^{(l+1)} = \mathbf{u}_i^{(l)} + \mathbf{v}_i^{(l+1)}$$

- *d) Termination*: If maximum number of iterations $I_{\max}$ is reached, terminate with solution $\mathbf{gb}^{(I_{\max})}$; otherwise, $l = l + 1$ and goto *b)*

Algorithm details can be found in the Proceeding

PSO Algorithmic Parameters

- Inertial weight $w_{\mathrm{I}} = rand()$, other alternative is $w_{\mathrm{I}} = 0$ or w_{I} set to a small positive constant
- Time varying acceleration coefficients

$$c_1 = (0.5 - 2.5) * l / I_{\max} + 2.5, \; c_2 = (2.5 - 0.5) * l / I_{\max} + 0.5$$

  - Initially, large cognitive component and small social component help particles to exploit better search space
  - Later, small cognitive component and large social component help particles to converge quickly to a minimum
- $S = 10$ to 20 appropriate for small to medium size problems, and empirical results suggest $I_{\max} = 20$ is often sufficient
- Search space is specified by problem, velocity space can be determined with $V_{j,\max} = 0.5 * (U_{j,\max} - U_{j,\min})$

Computational Complexity

- Let complexity of evaluating cost function once be $C_{\text{single}} \Rightarrow$ total complexity in determining one RBF node is

$$C_{\text{total}} = I_{\max} \times S \times C_{\text{single}}$$

  - Complexity of one LOO cost evaluation and associated column orthogonalisation is order of $K \Rightarrow C_{\text{single}} = \mathcal{O}(K)$

- Complexity of PSO-aided OFR in constructing M tunable-bases

$$C_{\text{PSO-OFR}} = (M+1) \times I_{\max} \times S \times \mathcal{O}(K)$$

- Complexity of LROLS-LOO in selecting $M^{'}$ fixed-bases from K-candidate set is

$$C_{\text{LROLS}} = \left(M^{'}+1\right) \times K \times \mathcal{O}(K)$$

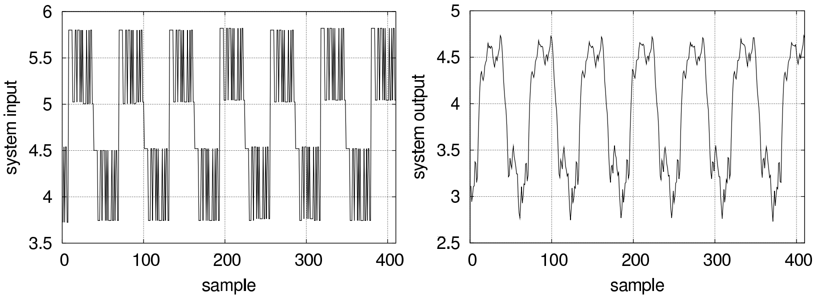
- PSO-aided OFR is generally simpler for large data set:

  $M < M^{'}$, typically $I_{\max} \times S \leq 400$: when $K \geq 400$, $C_{\text{PSO-OFR}} < C_{\text{LROLS}}$

Outline

Engine Data

- Data collected from a Leyland TL11 turbocharged, direct injection diesel engine operated at low engine speed
- System input u_k is fuel rack position, and system output y_k is engine speed



- First 210 data points for training, and last 200 data for testing

Experiment Results

- Training data $\{\mathbf{x}_k, y_k\}_{k=1}^{K}$ with $K = 210$ and

$$\mathbf{x}_k = [y_{k-1} \ u_{k-1} \ u_{k-2}]^T$$

- LROLS-LOO for fixed-node RBF model: every $\mathbf{x}_k$ as RBF centre, and RBF variance $\sigma^2 = 1.69$ determined via cross validation

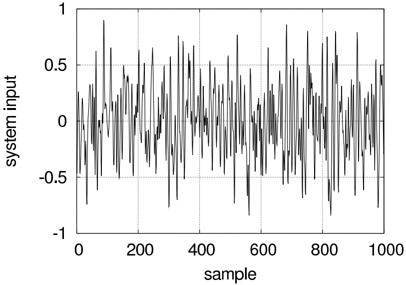
- PSO aided OFR for tunable-node RBF model: $S = 10$ and $I_{\max} = 20$

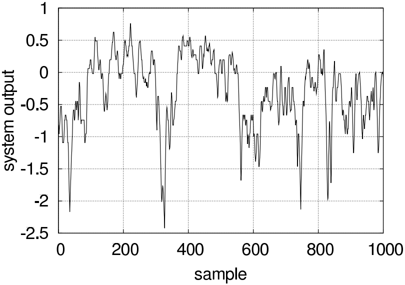
algorithm	model size	training MSE	test MSE	complexity
LROLS	22	0.000453	0.000490	$4830 \times \mathcal{O}(210)$
PSO OFR	15	0.000426	0.000466	$3200 \times \mathcal{O}(210)$

Outline

Liquid Level System Data

- Nonlinear liquid level system consists of a DC water pump feeding a conical flask which in turn feeds a square tank
- System input u_k is voltage to pump motor, and system output y_k is water level in conical flask





- First 500 data points for training, and last 500 data for testing

Experiment Results

- Training data $\{\mathbf{x}_k, y_k\}_{k=1}^{K}$ with $K = 500$ and

$$\mathbf{x}_k = [y_{k-1}\ y_{k-2}\ y_{k-3}\ u_{k-1}\ u_{k-2}\ u_{k-3}\ u_{k-4}]^T$$

- LROLS-LOO for fixed-node RBF model: every $\mathbf{x}_k$ as RBF centre, and RBF variance $\sigma^2 = 2.0$ determined via cross validation
- PSO aided OFR for tunable-node RBF model: $S = 10$ and $I_{\max} = 20$

algorithm	model size	training MSE	test MSE	complexity
LROLS	30	0.001400	0.002532	$15500 \times \mathcal{O}(500)$
PSO OFR	20	0.001461	0.002463	$4200 \times \mathcal{O}(500)$

Conclusions

We have developed a PSO aided OFR-LOO algorithm for constructing tunable-node RBF models.

- It combines advantages of linear and nonlinear learning.

Compared with the best algorithm for selecting subset model from the full fixed-node candidate set,

- it offers better test performance, smaller model size, and lower complexity in model construction process.