AR MODEL PARAMETER ESTIMATION: FROM FACTOR GRAPHS TO ALGORITHMS

Sascha Korl, Hans-Andrea Loeliger

ETH Zürich
Signal and Information Processing Laboratory
{korl,loeliger}@isi.ee.ethz.ch

Allen G. Lindgren

University of Rhode Island
Dept. of Electr. & Comp. Eng.
lindgren@ele.uri.edu

ABSTRACT

The classic problem of estimating the parameters of an auto-regressive (AR) model is considered from a graphical-model viewpoint. A number of practical parameter estimation algorithms—some of them well known, others apparently new—are derived as "summary propagation" in a factor graph. In particular, we demonstrate joint estimation of AR coefficients, innovation variance, and noise variance.

1. INTRODUCTION

Factor graphs are graphical models with an origin in the theory of error correcting codes. A wide variety of algorithms in coding, signal processing, and artificial intelligence may be viewed as "message passing" algorithms in graphical models such as factor graphs [1, 2]. In this paper, we use factor graphs to derive message passing algorithms for parameter estimation of Gaussian AR models. In particular, we demonstrate joint estimation of the AR parameters, of the innovation variance, and of the noise variance by algorithms that may be viewed as combinations of Kalman filters, LMS-type algorithms, and particle filters.

Specifically, we consider the following problem. Let $X_1, X_2, \ldots$ be a random process defined by

$$X_n = a_1 X_{n-1} + a_2 X_{n-2} + \cdots + a_M X_{n-M} + U_n \quad (1)$$

with unknown real parameters $a_1, \ldots, a_M$ and where $U_1, U_2, \ldots$ is zero-mean white Gaussian noise with variance σ_U^2. We observe

$$Y_n = X_n + W_n, \quad (2)$$

where W_n is zero-mean white Gaussian noise with variance σ_W^2. For later reference, we write (1) and (2) in state-space form as

$$\mathbf{X}_n = \mathbf{A}\mathbf{X}_{n-1} + \mathbf{b}U_n \quad (3)$$

$$Y_n = \mathbf{c}^T \mathbf{X}_n + W_n \quad (4)$$

with

$$\mathbf{X}_n = [X_n, \ldots, X_{n-M+1}]^T \quad (5)$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}^T \\ \mathbf{I} \quad \mathbf{0} \end{bmatrix} \quad (6)$$

$$\mathbf{b} = \mathbf{c} = [1, 0, \ldots, 0]^T \quad (7)$$

$$\mathbf{a} = [a_1, \ldots, a_M]^T. \quad (8)$$

From the observations $[Y_1, \ldots, Y_N] = [y_1, \ldots, y_N]$ we wish to estimate the AR parameters $a_1, \ldots, a_M$. In a first problem, the variances σ_U^2 and σ_W^2 are known; in a second problem, these variances are unknown and need to be estimated as well.

2. FACTOR GRAPH

The factor graph for our system model is depicted in Fig. 1. The figure shows only one section ("time slice") of the graph; the total graph consists of many such sections, one for each time index n.

The basic state space model is represented by the part consisting of solid lines in the middle. The (unknown) AR coefficient vector $\mathbf{a}$ is represented by the dashed edges and the dotted edges represent the variance of the innovation σ_U^2 and the variance of the noise σ_W^2, respectively.

In general, a Forney-style factor graph [2] represents a factorisation of a function of several variables. It consists of nodes, edges and half-edges, which are connected to only one node, and there are the following rules:

- There is a node for every factor.
- There is an edge (or half-edge) for every variable.
- The node representing some factor g is connected with the edge (or half-edge) representing some variable x if and only if x is an argument of g.

The restriction that no variable appears in more than two factors is easily circumvented by variable cloning. The single factors of the factorisation are also called *local functions*, the overall function (i.e. the product of all local functions) is called the *global function*. For our example, the

Sascha Korl is supported by KTI/CTI.

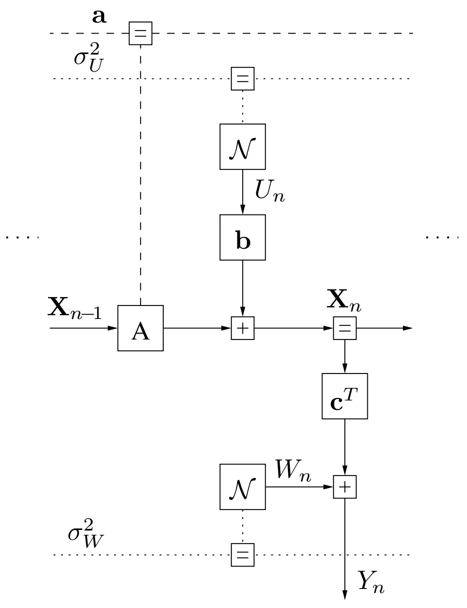

Fig. 1. Factor graph corresponding to state space model.

global function is the likelihood function

$$p(x_1,..,x_N,y_1,..,y_N,u_1,..,u_N,w_1,..,w_N|\mathbf{a},\sigma_U^2,\sigma_W^2). \tag{9}$$

An example of a local function is $p(y_1|x_1,w_1)$ for the addition node at the bottom.

Only the variables $[Y_1,\ldots,Y_N]=[y_1,\ldots,y_N]$ are observed. Estimates are determined from the posterior distributions of the parameters, which are obtained by marginalising the function (9). Such marginals can be computed—exactly or (if the graph has cycles) approximately—by message passing in the factor graph [1, 2].

Each message is a summary of the graph "behind" it and is a function of the variable that is represented by the corresponding edge. These functions may be represented in various ways, for example in parametric form for Gaussians or as a list of samples in Monte Carlo methods.

3. MESSAGE COMPUTATION

We briefly outline the role of all the messages in Fig. 2, and then we will describe their computation.

The computation of messages 1 to 5 amounts to standard Kalman filtering. Messages 6 and 7 arise in standard Kalman smoothing. Messages 8 and 9 complete the estimate of the AR coefficients; we will give two versions for these messages, one corresponding to an RLS (recursive least squares) algorithm and the other corresponding to an LMS (least mean square) algorithm. Messages 10 to 13 (and similarly 14 to 17) accomplish the variance estimation and will be computed by Monte Carlo methods.

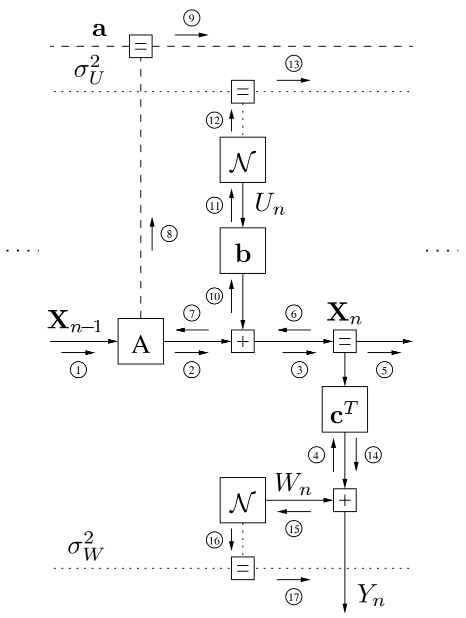

Fig. 2. Messages as discussed in Section 3.

The messages will be denoted by $\mu_\ell(x)$, where the number ℓ identifies the message in Fig. 2. We will now sketch the computation of these messages for $\ell=1,2,\ldots,17$. We will often use the update formulas for the standard building blocks listed in [3].

1. State estimate at time $n-1$ (computed at the previous time step):

$$\mu_1(\mathbf{x}_{n-1})=\mathcal{N}(\mathbf{x}_{n-1}\mid\hat{\mathbf{x}}_{n-1|n-1},\mathbf{V}_{n-1|n-1}) \tag{10}$$

where $\mathcal{N}(\mathbf{x}|\mathbf{m},\mathbf{V})$ denotes a Gaussian distribution in $\mathbf{x}$ with mean vector $\mathbf{m}$ and covariance matrix $\mathbf{V}$. The index $n-1|n-1$ refers to an estimate at time $n-1$ given all observations up to time $n-1$.

2. Forward matrix multiplication (with the coefficient vector $\hat{\mathbf{a}}_{n-1}$ from the previous time step):

$$\mathcal{N}(\,.\mid\hat{\mathbf{A}}_{n-1}\hat{\mathbf{x}}_{n-1|n-1},\hat{\mathbf{A}}_{n-1}\mathbf{V}_{n-1|n-1}\hat{\mathbf{A}}_{n-1}^T) \tag{11}$$

3. Accounting for the innovation:

$$\mu_3(\mathbf{x}_n)=\mathcal{N}(\mathbf{x}_n\mid\hat{\mathbf{x}}_{n|n-1},\mathbf{V}_{n|n-1}) \tag{12}$$

$$\hat{\mathbf{x}}_{n|n-1}=\hat{\mathbf{A}}_{n-1}\hat{\mathbf{x}}_{n-1|n-1} \tag{13}$$

$$\mathbf{V}_{n|n-1}=\hat{\mathbf{A}}_{n-1}\mathbf{V}_{n-1|n-1}\hat{\mathbf{A}}_{n-1}^T+\mathbf{b}\hat{\sigma}_{U_{n-1}}^2\mathbf{b}^T \tag{14}$$

These are the classic Kalman filter equations for the prediction step.

4. Observation afflicted by measurement noise:

$$\mathcal{N}(\,.\mid y_n,\hat{\sigma}_{W_{n-1}}^2) \tag{15}$$

5. Update with new information from observation (using the matrix inversion lemma):

$$\mu_5(\mathbf{x}_n) = \mathcal{N}(\mathbf{x}_n \mid \hat{\mathbf{x}}_{n|n}, \mathbf{V}_{n|n}) \tag{16}$$

$$\hat{\mathbf{x}}_{n|n} = \hat{\mathbf{x}}_{n|n-1} + \frac{\mathbf{V}_{n|n-1}\mathbf{c}\ (y_n - \mathbf{c}^T \hat{\mathbf{x}}_{n|n-1})}{\hat{\sigma}^2_{W_{n-1}} + \mathbf{c}^T \mathbf{V}_{n|n-1}\mathbf{c}} \tag{17}$$

$$\mathbf{V}_{n|n} = \mathbf{V}_{n|n-1} - \frac{\mathbf{V}_{n|n-1}\mathbf{c}\ \mathbf{c}^T \mathbf{V}_{n|n-1}}{\hat{\sigma}^2_{W_{n-1}} + \mathbf{c}^T \mathbf{V}_{n|n-1}\mathbf{c}} \tag{18}$$

These are the classic Kalman filter equations for the update step.

To update the estimate of the AR coefficients $\mathbf{a}$, information from both the old state and the new observation has to be integrated. Messages 4 (already computed), 6 and 7 propagate the observation back to the state transition node $\mathbf{A}$. In the first iteration, there is no information about some entries in the state vector; this is reflected by the $\mathbf{I}_\infty$ in the covariance matrix, which has ∞'s in its diagonal and is zero elsewhere.

6.

$$\mu_6(\mathbf{x}_n) = \mathcal{N}(\mathbf{x}_n \mid \mathbf{m}_6, \mathbf{V}_6) \tag{19}$$

$$\mathbf{m}_6 = \begin{pmatrix} y_n \\ \mathbf{0} \end{pmatrix} \quad \mathbf{V}_6 = \begin{bmatrix} \hat{\sigma}^2_{W_{n-1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_\infty \end{bmatrix} \tag{20}$$

From the second iteration on, this result is modified by the message arriving from the right side.

7. Accounting for the innovation:

$$\mu_7(.) = \mathcal{N}(\,.\mid \mathbf{m}_7, \mathbf{V}_7) \tag{21}$$

$$\mathbf{m}_7 = \begin{pmatrix} y_n \\ \mathbf{0} \end{pmatrix} \quad \mathbf{V}_7 = \begin{bmatrix} \hat{\sigma}^2_{W_{n-1}} + \hat{\sigma}^2_{U_{n-1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_\infty \end{bmatrix} \tag{22}$$

The exact form of message 8 is not Gaussian anymore but too unwieldy to be processed further. Two remedies are proposed: either applying a suitable approximation to the true distribution (for example by moment matching as in [4]) or resort to search methods such as gradient-based methods.

Ignoring the variance in the state estimate $\hat{\mathbf{x}}_{n-1|n-1}$ leads to a Gaussian message for 8. This message can be processed by the usual building blocks.

8. (a) The mean and weight matrix for this message:

$$\mathbf{m}_8 = \frac{y_n}{\|\hat{\mathbf{x}}_{n-1|n-1}\|^2}\hat{\mathbf{x}}_{n-1|n-1} \tag{23}$$

$$\mathbf{W}_8 = \frac{\hat{\mathbf{x}}_{n-1|n-1}\hat{\mathbf{x}}^T_{n-1|n-1}}{\hat{\sigma}^2_{W_{n-1}} + \hat{\sigma}^2_{U_{n-1}}} \tag{24}$$

9. (a) The update at the equality node propagating the coefficient estimate is:

$$(\mathbf{W}_\mathbf{a})_n = (\mathbf{W}_\mathbf{a})_{n-1} + \frac{\hat{\mathbf{x}}_{n-1|n-1}\hat{\mathbf{x}}^T_{n-1|n-1}}{\hat{\sigma}^2_{W_{n-1}} + \hat{\sigma}^2_{U_{n-1}}} \tag{25}$$

$$\hat{\mathbf{a}}_n = \hat{\mathbf{a}}_{n-1} + \frac{(\mathbf{W}_\mathbf{a})_n^{-1}\hat{\mathbf{x}}_{n-1|n-1}}{\hat{\sigma}^2_{W_{n-1}} + \hat{\sigma}^2_{U_{n-1}}} e_n \tag{26}$$

$$e_n = y_n - \hat{\mathbf{a}}^T_{n-1}\hat{\mathbf{x}}_{n-1|n-1} \tag{27}$$

For $\sigma^2_W = 0$ this is the classic RLS formulation of AR parameter estimation.

Instead of computing the message in closed form, one may compute its gradient. The actual estimate of the coefficients is improved by iterative updating [2].

8. (b) In deriving the gradient, approximations can be made without considerably affecting the performance of the resulting algorithm. In the simplest case this leads to

$$\nabla_\mathbf{a} \log \mu_8(\mathbf{a}) \approx 2\hat{\mathbf{x}}_{n-1|n-1}(y_n - \mathbf{a}^T \hat{\mathbf{x}}_{n-1|n-1}) \tag{28}$$

9. (b) The update of the coefficients is then

$$\hat{\mathbf{a}}_n = \hat{\mathbf{a}}_{n-1} + s\ \hat{\mathbf{x}}_{n-1|n-1} e_n \tag{29}$$

with e_n given by (27) and stepsize s. This resembles the classic LMS algorithm.

This completes the solution to our first problem, the estimation of the AR coefficient vector $\mathbf{a}$. In the second problem the innovation and noise variances are unknown and have to be estimated as well, which is carried out by messages 10 to 13 for the innovation variance and 14 to 17 for the noise variance, respectively.

10. The difference of forward and backward estimates:

$$\mu_{10}(.) = \mathcal{N}(\,.\mid \mathbf{m}_{10}, \mathbf{V}_{10}) \tag{30}$$

$$\mathbf{m}_{10} = \mathbf{m}_6 - \hat{\mathbf{A}}_{n-1}\hat{\mathbf{x}}_{n-1|n-1} \tag{31}$$

$$\mathbf{V}_{10} = \hat{\mathbf{A}}_{n-1}\mathbf{V}_{n-1|n-1}\hat{\mathbf{A}}^T_{n-1} + \mathbf{V}_6 \tag{32}$$

11. The estimate of the input:

$$\mu_{11}(u_n) = \mathcal{N}(u_n \mid m_{11}, V_{11}) \tag{33}$$

$$m_{11} = y_n - \hat{\mathbf{a}}^T_{n-1}\hat{\mathbf{x}}_{n-1|n-1}. \tag{34}$$

where V_{11} is the variance as the result of the backward matrix multiplication, which reduces to the element in the first row and first column of the covariance matrix in (32).

The next step—computing message 12—involves again a message which is non-Gaussian:

$$\mu_{12}(\sigma^2) = \frac{1}{\sqrt{2\pi(\sigma^2 + V_{11})}} \exp\left(-\frac{m_{11}^2}{2(\sigma^2 + V_{11})}\right)$$
$$\propto \mathrm{Ig}\left(\sigma^2 + V_{11}\middle| -\frac{1}{2}, \frac{m_{11}^2}{2}\right) \tag{35}$$

where $\mathrm{Ig}(.|\alpha, \beta)$ denotes an inverted gamma distribution with parameters α and β, but shifted by the amount of V_{11}. Again, since this distribution cannot be processed further in a practical way (it is not closed under multiplication), we have to resort to different methods such as gradient based methods shown above or Monte Carlo simulation methods, especially particle methods. The general idea behind particle methods is to represent a probability distribution by a list of samples. Either the messages itself are such lists and are weighted/resampled by the nodes [5] or the messages are approximated by Gaussians and particles are only used to compute the update equations inside each node.

12. & 13. Use one of the above mentioned methods to generate either an assumed Gaussian message

$$\mu_{13}(\sigma^2_{U_n}) = \mathcal{N}\left(\sigma^2_{U_n} \,\middle|\, \hat{\sigma}^2_{U_n}, V_{13}\right) \tag{36}$$

or a list of samples representing the true distribution.

14. Predicted clean output:

$$\mathcal{N}\left(\,.\, \middle| \, \hat{\mathbf{a}}^T_{n-1}\hat{\mathbf{x}}_{n-1|n-1}, \mathbf{c}^T\mathbf{V}_{n|n-1}\mathbf{c} + \hat{\sigma}^2_{U_{n-1}} \right) \tag{37}$$

15. The estimate of the noise:

$$\mu_{15}(w_n) = \mathcal{N}(w_n \mid m_{15}, V_{15}) \tag{38}$$

$$m_{15} = y_n - \hat{\mathbf{a}}^T_{n-1}\hat{\mathbf{x}}_{n-1|n-1} \tag{39}$$

Messages 16 and 17 may be computed in the same way as messages 12 and 13.

4. MESSAGE UPDATE SCHEDULE

In the case when $\sigma^2_W = 0$ and σ^2_U is fixed and known, a forward-only (left-to-right) message update schedule works fine. In this case, the proposed message passing algorithm reduces either to the standard RLS algorithm (when (25)–(27) are used) or to the LMS algorithm (when (27)–(29) are used). In the more interesting cases ($\sigma^2_W \neq 0$, known or unknown), however, forward-only message propagation performs poorly. We then use iterative message updating, with several rounds of computing, first, the messages 1, 2, 3, 4, 5, 14, 15, 16, 17, 6, 10, 11, 12, 13, 7, 8, 9 from left to right and then the messages 4, 6, 7, 8 from right to left.

5. SIMULATION RESULTS

As a demonstration, we show the estimation errors for an 8th order AR model with the following parameters: $\sigma^2_U = 0.1$, $\sigma^2_W = 0.01/0.001$, $\mathbf{a} = [1.51, \text{-}1.08, 0.47, \text{-}0.23, 0.91, \text{-}1.30, 0.86, \text{-}0.32]^T$, $N = 10^4$.

Fig. 3(a) shows relative errors of the forward estimates vs. slice index k for both the new iterative estimator (after 3 iterations) and the classical LPC using the RLS algorithm. The standard RLS estimator with no measurement noise yields an error that goes to zero as $1/k$. It is well known that the RLS algorithm, when used with noise corrupted observations, produces biased estimates, as can be seen in Fig. 3(a) (dashed lines). The new estimator is seen to produce coefficient error levels that decrease as $1/k$ also in the noisy case.

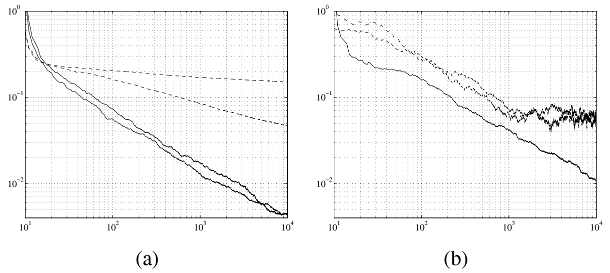

Fig. 3. Averaged relative errors of (a) coefficients for the new iterative estimator (solid) and classical RLS-LPC (dashed) with $\sigma^2_U = 0.1$ and $\sigma^2_W = 0.01/0.001$ (upper/lower curve) and of (b) coefficients (solid), innovation variance (dash-dot) and noise variance (dashed) for the joint estimation problem vs. slice k.

Estimation errors for the joint estimation problem are shown in Fig. 3(b) after the fifth iteration. The flattening of the variance estimates at $6 \cdot 10^{-2}$ is due to the limited resolution of the particle filter; methods to overcome this limitation will be discussed elsewhere.

6. CONCLUSIONS

We have shown how estimation algorithms for the parameters of an AR model can be derived from a factor graph. In particular, we have demonstrated joint estimation of the AR parameters, the innovation variance, and the noise variance. Some noteworthy features of the graphical-model approach to such problems are the following:

- Different signal processing techniques such as Kalman filtering, gradient methods, and particle filters can easily be combined.
- An essential complexity reduction is achieved by factored representations of the overall state space, which amounts to factor graphs with cycles.
- Extensions from simple models to more complex models are often easily achieved. For example, it is not difficult to extend the estimation algorithms of this paper to time-varying models.

7. REFERENCES

[1] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product-algorithm," *IEEE Trans. on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[2] H.-A. Loeliger, "An introduction to factor graphs," *IEEE Signal Processing Mag.*, Jan. 2004.

[3] H.-A. Loeliger, "Least squares and Kalman filtering on Forney graphs," in *Codes, Graphs, and Systems*, R.E. Blahut and R. Koetter, Eds. Kluwer, 2002.

[4] T. P. Minka, *A Family of Algorithms for Approximate Bayesian Inference*, Ph.D. thesis, MIT, 2001.

[5] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, 2001.