Research Article
Cost-Sensitive Estimation of ARMA Models for Financial Asset Return Data

Minyoung Kim

Department of Electronics & IT Media Engineering, Seoul National University of Science & Technology, Seoul 139-743, Republic of Korea

Correspondence should be addressed to Minyoung Kim; mikim21@gmail.com

Received 11 June 2015; Revised 30 August 2015; Accepted 2 September 2015

Academic Editor: Meng Du

Copyright © 2015 Minyoung Kim. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The autoregressive moving average (ARMA) model is a simple but powerful model in financial engineering to represent time-series with long-range statistical dependency. However, the traditional maximum likelihood (ML) estimator aims to minimize a loss function that is inherently symmetric due to Gaussianity. The consequence is that when the data of interest are asset returns, and the main goal is to maximize profit by accurate forecasting, the ML objective may be less appropriate potentially leading to a suboptimal solution. Rather, it is more reasonable to adopt an asymmetric loss where the model’s prediction, as long as it is in the same direction as the true return, is penalized less than the prediction in the opposite direction. We propose a quite sensible asymmetric cost-sensitive loss function and incorporate it into the ARMA model estimation. On the online portfolio selection problem with real stock return data, we demonstrate that the investment strategy based on predictions by the proposed estimator can be significantly more profitable than the traditional ML estimator.

1. Introduction

In modeling time-series data, capturing the underlying statistical dependency of the variables of interest at current time on the historic data is central to accurate forecasting and faithful data representation. For financial time-series data especially (e.g., daily asset prices or returns) where a large amount of potential prognostic indicators is available, the development/analysis of sensible dynamic models as well as effective parameter estimation algorithms has been investigated significantly.

To account for statistical properties specific to financial sequences, several sophisticated dynamic time-series models have been developed: fairly natural autoregressive and/or moving average models [1], the conditional heteroscedastic models that represent dynamics of volatilities (variances) of the asset returns [2–4], and nonlinear models [5, 6] including bilinear models [7], threshold models [8], and regime switching models [9, 10].

Among those, the autoregressive moving average (ARMA) model [1] is the simplest while essential in the sense that most other models are equipped with at least the basic ARMA components. The ARMA models appear in a wide spectrum of applications recently including filter design in signal processing [11], time-series analysis and model selection in computational statistics [12], and jump (large changes) modeling for asset prices in quantitative finance [13], to name just a few. For a time-series $y = y_1, \ldots, y_T$ (e.g., $y_t$ is the asset return at the $t$th day), the ARMA($p,q$) determines $y_t$ by

$$y_t = \alpha^T y_{t-p} + \beta^T e_{t-q} + \beta_0 e_t + \rho,$$

where $x_T^T$ indicates the vector $[x_{t-p}, x_{t-q}, \ldots, x_t]^T$. Here $e_t \sim \mathcal{N}(0, \sigma^2)$ is the stochastic Gaussian error term at time $t$ we assume iid across $t$‘s. In (1) $\alpha, \beta, \beta_0,$ and $\rho$ are the model parameters. That is, $y_t$ is dependent on $p$ previous asset returns, $q$ historic errors, and the current error $e_t$.

In this paper we consider a more general, recent stochastic extension of ARMA (abbreviated as sARMA) [14] (in contrast to the deterministic equation (1)) that adds a Gaussian noise to (1). Moreover, the extra covariates $c_t$ (called cross predictors) are assumed available at time $t$; for instance, they are typically economic indicators, market indices, and/or the previous...
returns of other related assets. The sARMA(p, q) model can be specifically written as
\[
y_t \mid y_{t-p}, e_{t-q}, c_t \sim \mathcal{N}(u_t, \sigma_t^2),
\]
where
\[
u_t = \alpha^T y_{t-1} + \beta^T e_{t-q} + \gamma^T c_t + \rho.
\]
Here \(\eta\) is the weight vector (model parameters) for the cross predictor.

Hence, sARMA deals with Gaussian noisy observation (with variance \(\sigma^2\)), and it exactly reduces to the ARMA model in the limiting case \(\sigma \to 0\). The noisy observation modeling of sARMA is beneficial in several aspects: not only does it merely account for the underlying noise process in the observation but also the model becomes fully stochastic, which allows principled probabilistic inference and model estimation even with missing data [14].

Given the observed sequence data, the parameters \(\theta = \{\gamma^2, \alpha, \beta, \rho_0, \eta, \sigma^2\}\) of the sARMA model can be estimated by the expectation maximization (EM) algorithm [15]. Compared to the traditional Levenberg-Marquardt method for ARMA model estimation [1], the EM algorithm is beneficial for dealing with latent variables (i.e., the error terms) as well as any missing observations in an efficient and principled way. However, both estimators basically aim to achieve data likelihood maximization (ML) under the Gaussian model (2) [14] (with \(\sigma \to 0\) corresponding to ARMA).

Due to the Gaussian observation modeling in sARMA, the ML estimation inherently aims to minimize a symmetric loss. In other words, letting \(\hat{y}_t\) and \(\overline{y}_t\) be the model forecast and the true value at time \(t\), respectively, incorrect prediction \(\hat{y}_t\), with the prediction error \(\xi = |\hat{y}_t - \overline{y}_t|\) incurs the same amount of loss for both \(\hat{y}_t = \overline{y}_t + \xi\) and \(\overline{y}_t = \hat{y}_t - \xi\) (i.e., regardless of over- or underestimation). This strategy is far from being optimal especially for the asset return data as argued in the following.

The main goal is to maximize profit by accurate forecasting with the asset return data that encode signs (directions) toward profits. Traditional maximum likelihood (ML) estimator aims to minimize a loss function that is inherently symmetric and hence unable to exploit the property of the asset return data, leading to a suboptimal solution.

Suppose that our data \(y\) forms a sequence of daily stock log-returns, encoded as \(y_t > 0\) (\(<0\)) indicating that the stock price moves up (declines) on the \(t\)th day against the previous day. Now, consider a portfolio selection algorithm that makes an investment based on the forecast \(\hat{y}_t\) given the information up to time \(t\). The investment yields positive revenue when the signs of \(\hat{y}_t\) and \(\overline{y}_t\) are equal and the other way around. Hence, the prediction loss should be inherently asymmetric. Furthermore, when \(\overline{y}_t < 0\), having \(\hat{y}_t < 0\), even its underestimation (i.e., \(\hat{y}_t < \overline{y}_t\)) should be penalized less than the prediction with the opposite direction (i.e., \(\hat{y}_t > 0\)) because the former does not incur any loss in revenue but the latter does.

To address this issue, we propose a reasonable cost function that effectively captures the above idea of the intrinsic asymmetric profit/loss structure regarding asset return data. Our cost function effectively encodes the goodness of matching in directions between true and model predicted asset returns, which is directly related to ultimate profits in the investment. We also provide an efficient optimization strategy based on the subgradient descent using the trust-region approximation, whose effectiveness is empirically demonstrated for the portfolio selection problem with real-world stock return data.

It is worth mentioning that there have been several other asymmetric loss functions proposed in the literature similar to ours. However, existing loss models merely focus on the asymmetry with respect to the ground-truth value point. For instance, the linex function [16, 17] is defined to be a linear-exponential function of difference between predicted and ground-truth values. The linlin method [18] adopts a piecewise linear function where the change point is simply the ground-truth value. To the best of our knowledge, we are the first to derive the loss based on the matching the directions (signs) of the predicted and ground-truth returns. This effectively enables incorporating the critical information about directions of profits/losses, in turn leading to a more accurate forecasting model.

The rest of the paper is organized as follows. In the next section we suggest a novel sARMA estimation algorithm based on the cost-sensitive loss function: beginning with the overall objective, we derive the one-step predictor for the sARMA model in Section 2.1, provide details of the proposed cost function in Section 2.2, and state the optimization strategy in Section 2.3. The statistical inference algorithm for the sARMA model is also provided in full derivations in Section 2.4. In the empirical study in Section 3, we demonstrate the effectiveness of the proposed algorithm on the online portfolio selection problem with real data, where the significantly higher total profit is attained by the proposed approach than the investment based on the traditional ML-estimated sARMA model.

2. Cost-Sensitive Estimation

The proposed estimator for sARMA is based on the cost-sensitive loss of the model predicted one-step forecast value (denoted by \(\hat{y}_t\)) at each time \(t\) with respect to the true one (denoted by \(\overline{y}_t\)) available from data. More specifically, for a given data sequence \(y = y_1, \ldots, y_T\), we aim to solve the optimization problem:

\[
\min_{\theta} \sum_{t=1}^{T} C(\hat{y}_t, \overline{y}_t) + \lambda N(\theta),
\]

where \(R = \max(p, q)\).

Here \(C(\hat{y}_t, \overline{y}_t)\) is the cost of predicting the asset return as \(\hat{y}_t\) when the true value is \(\overline{y}_t\). In Section 2.2 we define a reasonable cost function that faithfully incorporates the idea of asymmetric cost-sensitive loss discussed in the introduction.

In the objective, we also simultaneously minimize \(N(\theta)\), the parameter regularizer that typically penalizes a nonsmooth sARMA model while preferring a smooth model (effectively achieved by encouraging the regression
Mathematical Problems in Engineering

parameters in \( \theta \) close to 0) model. Specifically we use the L2 penalty, \( N(\theta) = \| \alpha \|^2 + \| \beta \|^2 + \| \eta \|^2 \). The constant \( A > 0 \) trades off the regularisation against the prediction error cost.

Note also that in (4) we use the notation \( \hat{y}_t(\theta) \) to emphasize the dependency of the model predicted \( \hat{y}_t \) on \( \theta \). We use the principled maximum a posteriori (MAP) predictor estimated under the sARMA model, which is fully described in Section 2.1. The predictor is evaluated based on the inference on the latent error terms, which can be computed recursively where we give detailed derivations for the inference in Section 2.4.

2.1. One-Step Predictor for sARMA. Under the sARMA model, the predictive model at time \( t \), given all available information \( (y_t^{1−1}, e_t^{R}, c_t') \), is \( P(y_t | y_t^{1−1}, e_t^{R}, c_t') \) for \( t = R + 1, R + 2, \ldots \). From this predictive model, one can make deterministic decision on the asset return at \( t \), typically as the maximum-a-posteriori (MAP) estimation:

\[
\hat{y}_t = \arg \max_{y_t} P(y_t | y_t^{1−1}, e_t^{R}, c_t').
\] (5)

Note that in the sARMA model, it is always assumed that we have at least \( R \) previous observations \( y_t^{R} \) and \( R \) previous error terms \( e_t^{R} \). The error terms are simply assumed to be \( e_1 = \ldots = e_R = 0 \) throughout the paper. Due to the linear Gaussianity of the sARMA's conditional densities, we have Gaussian \( P(y_t | y_t^{1−1}, e_t^{R}, c_t') \), and the MAP predictor (5) exactly coincides with the mean \( \mathbb{E}[y_t | y_t^{1−1}, e_t^{R}, c_t'] \).

In this section we derive the MAP (or mean) prediction \( \hat{y}_t \) as a function of the sARMA model parameters \( \theta \), which can then be used in gradient evaluation for the optimization in (4). As is shown, the predictive distributions heavily resort to the posterior distributions of the error terms, namely, \( P(e_{R+1}^t | y_t^{1−1}, e_t^{R}, c_t') \) for \( t = R + 1, \ldots, T \). They are also Gaussians, and we denote them by \((\mu_t, \Sigma_t)\) in

\[
P(e_{R+1}^t | y_t^{1−1}, e_t^{R}, c_t') = N(\mu_t, \Sigma_t),
\] (6)

for \( t = R + 1, \ldots, T \). Note that \( \mu_t \) and \( \Sigma_t \) have dimensions \((t−R) \times 1\) and \((t−R) \times (t−R)\), respectively. The full derivation of the error term posteriors is provided in Section 2.4.

In deriving \( P(y_t | y_t^{1−1}, e_t^{R}, c_t') \), one may need to differentiate three cases for \( t \): (i) \( t = R + 1 \), (ii) \( R + 1 < t \leq R + q + 1 \), and (iii) \( t > R + q + 1 \). The first case simply forms the initial condition which immediately follows from the local conditional model with marginalization of \( e_{R+1} \). That is, when \( t = R + 1 \),

\[
P(y_{R+1} | y_t^{1−1}, e_t^{R}, c_t^{R+1}) = N(b_{R+1}, \beta_{R+1}^2 \gamma^2 + \sigma^2),
\] (7)

where we define \( b_t = \alpha^T y_{t-p} + \eta^T c_t + \rho \) for \( t = R + 1, \ldots, T \).

In the second and third cases for the following reason: at time \( t \), the previous \( q \) error terms are fully included in the time window \([R+1, t]\) in the latter case, while they are partially included in the former. Hence in the second case, we additionally deal with the error terms \( e_{t−q}^t \) which are always given as 0. Specifically, in the second case \((R + 1 < t \leq R + q + 1)\), the terms \( e_{t−q}^t \) are partitioned into \((e_t, e_{R+1}^R, e_{t−q}^R = 0)\), and we have

\[
P(y_t | y_t^{1−1}, e_t^{R}, c_t') = \sum_{\epsilon_{R+1}} P(y_t, e_{R+1}^R | y_t^{1−1}, e_t^{R}, c_t')
\] (8)

\[
= \sum_{\epsilon_{R+1}} P(y_t | y_t^{1−1}, e_{t−q}^t, c_t') \cdot P(e_{R+1}^R | y_t^{1−1}, e_t^{R}, c_t').
\] (9)

\[
= \sum_{\epsilon_{R+1}} N(y_t; b_t + \beta^T e_{t−q}^t + \beta_0 e_t, \sigma^2) \cdot P(\epsilon_t)
\] (10)

\[
\cdot N(\epsilon_{R+1}; \Sigma_{R+1}, \Sigma_{R+1})
\] (11)

\[
= N(y_t; b_t + \beta^T e_{t−q}^t, \beta_{R+1}^2 \gamma^2 + \sigma^2).
\] (12)

In (12), we let \( \beta_t \) be the subvector of \( \beta \) corresponding to \( e_{R+1}^t \).

In the third case \((t > R + q + 1)\), we only need to deal with error terms \( e_{t−q}^t \), and the predictive density is derived as follows:

\[
P(y_t | y_t^{1−1}, e_t^{R}, c_t') = \sum_{\epsilon_{t−q}} P(y_t | y_t^{1−1}, e_{t−q}^t, c_t') \cdot P(\epsilon_t)
\] (13)

\[
\cdot P(\epsilon_{t−q} | y_t^{1−1}, e_t^{R}, c_t')
\] (14)

\[
= \sum_{\epsilon_{t−q}} N(y_t; b_t + \beta^T e_{t−q}^t + \beta_0 e_t, \sigma^2)
\] (15)

In (14), we introduce \( \mu_{t−q} \) and \( \Sigma_{t−q} \) as submatrices of \( \mu_t \) and \( \Sigma_t \), taking the indices from \((t−q)\) to \((t−1)\) only.

In summary, the one-step predictor \( \hat{y}_t(\theta) \) at \( t \) with all available information \((y_t^{1−1}, e_t^{R}, c_t')\) can be written as

\[
\hat{y}_t(\theta) = \begin{cases} 
\alpha^T \hat{y}_{t−1}^R + \eta^T c_t + \rho & \text{if } t = R + 1 \\
\alpha^T \hat{y}_{t−p}^R + \eta^T c_t + \rho + \beta^T \mu_{t−1} & \text{if } R + 1 < t \leq R + q + 1 \\
\alpha^T \hat{y}_{t−p}^R + \eta^T c_t + \rho + \beta^T \hat{\mu}_{t−1} & \text{if } t > R + q + 1. 
\end{cases}
\] (16)

Note here that the means of the error term posteriors \( \mu_{t−1} \) (and their subvectors \( \hat{\mu}_{t−1} \)) have also dependency on the model parameters \( \theta \).

2.2. Proposed Cost Function. In this section we propose a cost function \( C(\hat{y}_n, F) \) (used in (4)) that effectively encodes the intrinsic asymmetric profit/loss structure regarding asset return data. To meet the motivating idea discussed in Section 1, we deal with two outstanding cases: the case when
the true \( \tilde{Y}_t \) is positive and the case when \( \tilde{Y}_t \) is negative. In each case, we further consider a certain margin \( \varepsilon \) (small positive, e.g., \( \varepsilon = 0.005 \)), where observing \( \tilde{Y}_t > \varepsilon \) indicates positive return with high certainty; on the other hand, having \( 0 \leq \tilde{Y}_t \leq \varepsilon \) can be regarded differently as weak positivity and might be considered as noise. For the negative return, we have similar two regimes of different certainty levels.

We discuss the first case, \( \tilde{Y}_t > \varepsilon \). Depending on the value of \( \tilde{Y}_t \), the cost functional changes over the four intervals: (i) \( \tilde{Y}_t < 0 \) incurs the highest loss with a super-linear penalty along the magnitude of \( \tilde{Y}_t \) (we particularly choose a convex quadratic function), (ii) \( \tilde{Y}_t \geq \tilde{Y}_t \), that is, overestimation, should be penalized the least, and we opt for an increasing linear function with a small slope, (iii) \( \varepsilon \leq \tilde{Y}_t < \tilde{Y}_t \) is an underestimation, but the prediction has certainty greater than a margin and thus is penalized less (we choose a linear function with slope slightly higher than the second case), and (iv) \( 0 \leq \tilde{Y}_t < \varepsilon \) makes prediction in correct direction, but due to the weak certainty below the margin, we penalize it more severely than previous two regimes.

Our specific cost definition for \( \tilde{Y}_t > \varepsilon \) is as follows:

\[
C(\tilde{Y}_t, Y_t) = \begin{cases} 
\frac{1}{2} s_o \tilde{Y}_t^2 - s_o \tilde{Y}_t + g_t & \text{if } \tilde{Y}_t < 0 \\
-s_o (\tilde{Y}_t - \varepsilon) + g_0 & \text{if } 0 \leq \tilde{Y}_t < \varepsilon \\
s_{\text{in}} (\tilde{Y}_t - Y_t) & \text{if } \varepsilon \leq \tilde{Y}_t < Y_t \\
s_{\text{out}} (\tilde{Y}_t - Y_t) & \text{if } \tilde{Y}_t \geq Y_t \\
(\text{when } \tilde{Y}_t > \varepsilon) & 
\end{cases} \tag{17}
\]

where we choose the constants as follows: \( s_o = 4, s_0 = 2, s_{\text{in}} = 0.5, s_{\text{out}} = 0.3 \). To make the cost function continuous, we define two offsets as follows: \( g_t = -s_{\text{in}}(e - \tilde{Y}_t) \) and \( g_0 = g_0 + \varepsilon s_0 \).

In the case of \( \tilde{Y}_t < -\varepsilon \), we exactly penalize the prediction in the same way as the first situation. Specifically, the cost definition for \( \tilde{Y}_t < -\varepsilon \) is

\[
C(\tilde{Y}_t, Y_t) = \begin{cases} 
\frac{1}{2} s_o \tilde{Y}_t^2 + s_o \tilde{Y}_t + h_t & \text{if } \tilde{Y}_t > 0 \\
s_o (\tilde{Y}_t + \varepsilon) + h_0 & \text{if } -\varepsilon < \tilde{Y}_t \leq 0 \\
s_{\text{in}} (\tilde{Y}_t - Y_t) & \text{if } \tilde{Y}_t < Y_t \leq -\varepsilon \\
s_{\text{out}} (\tilde{Y}_t - Y_t) & \text{if } \tilde{Y}_t \leq Y_t \\
(\text{when } \tilde{Y}_t < -\varepsilon) & 
\end{cases} \tag{18}
\]

where the same constants are used, and the offsets are now set as \( h_0 = -s_{\text{in}}(e + \tilde{Y}_t) \) and \( h_1 = h_0 + \varepsilon s_0 \).

For the uncertain (within the margin \( \varepsilon \)) return, we still conform to the strategy of encouraging the same direction as the true return. In the case of \( 0 < Y_t \leq \varepsilon \), we assign small penalty for overestimation as long as it is in the correct direction, while rapidly growing quadratic loss for the prediction toward opposite direction. To summarize, the cost for \( 0 < Y_t \leq \varepsilon \) is

\[
C(\tilde{Y}_t, Y_t) = \begin{cases} 
\frac{1}{2} s_o \tilde{Y}_t^2 - s_o \tilde{Y}_t + k_1 & \text{if } \tilde{Y}_t < 0 \\
-s_o (\tilde{Y}_t - Y_t) & \text{if } 0 \leq \tilde{Y}_t < Y_t \\
s_{\text{out}} (\tilde{Y}_t - Y_t) & \text{if } \tilde{Y}_t \geq Y_t \\
(\text{when } 0 < Y_t \leq \varepsilon) & 
\end{cases} \tag{19}
\]

where we set (for continuity) \( k_1 = s_0 Y_t \). The other case of \(-\varepsilon \leq Y_t < 0 \) is similarly defined as

\[
C(\tilde{Y}_t, Y_t) = \begin{cases} 
\frac{1}{2} s_o \tilde{Y}_t^2 + s_o \tilde{Y}_t + l_1 & \text{if } \tilde{Y}_t > 0 \\
s_o (\tilde{Y}_t - Y_t) & \text{if } \tilde{Y}_t < Y_t \leq 0 \\
-s_{\text{out}} (\tilde{Y}_t - Y_t) & \text{if } \tilde{Y}_t \leq Y_t \\
(\text{when } -\varepsilon \leq Y_t < 0) & 
\end{cases} \tag{20}
\]

2.3. Optimization Strategy. In this section we briefly describe the optimization strategy for (4). We basically follow the subgradient descent [19, 20] where the derivative of the cost function with respect to \( \theta \) can be derived as

\[
\frac{\partial C(\tilde{Y}_t, Y_t)}{\partial \theta} = \frac{\partial C(\tilde{Y}_t, Y_t)}{\partial \tilde{Y}_t} \frac{\partial \tilde{Y}_t}{\partial \theta}.
\tag{21}
\]

Here, due to the nondifferentiability of the cost function (albeit continuous), we use the subgradient in place of the second part of RHS of (21).

Evaluating the first part, that is, the gradient of \( \tilde{Y}_t \) with respect to \( \theta \), requires further endeavor. According to the functional form of \( \tilde{Y}_t \) in (16), it has complex recursive dependency on \( \theta \) mainly due to the error posterior means \( \mu_{t-1} \). Instead of exactly computing the derivative of \( \mu_{t-1} \), we address this issue by evaluating an approximate gradient by treating \( \mu_{t-1} \) as a constant (constant evaluated at the current iterate \( \theta \)). In consequence, we have a linear function of \( \theta \), and the gradient can be computed easily. However, the approximation (i.e., constant \( \mu_{t-1} \) with respect to \( \theta \)) is only valid in the vicinity of the current \( \theta \). Hence, to reduce the approximation error, we restrict the search space to be not much different from the current iterate (i.e., we search the next \( \theta \) within the small-radius ball centered at the current \( \theta \), specifically \( ||\theta - \theta_{\text{curr}}|| \leq \kappa \) for some small \( \kappa > 0 \)). Our optimization strategy is closely related to the trust-region method [21], where the objective is approximated in the vicinity of the current parameters.

2.4. Inference in sARMA. In this section we give full derivations for statistical inference on the latent error variables \( e_{t+1}^{(t)} \) for each \( t \), conditioned on the historic observations \( Y_t \) and the cross predictors \( c_t \) in the sARMA model. That is, the posterior
densities \( P(e^t_{R+1} | y^t_1, e^R_1, c^R_1) \) for \( t = R + 1, \ldots, T \) are fully derived. In essence, these are all Gaussians, and as denoted in (6), we find the recursive formulas for the means and covariances \((\mu_t, \Sigma_t)\). We also denote the inverse covariance \(\Sigma^{-1}_t\) by \(S_t\).

Similarly, as one-step predictive distributions, we consider three cases: (i) initial \( t = R + 1 \), (ii) \( R + 1 < t \leq R + q + 1 \) where \( e_{t-q}^1 \) fully contains what we need to infer, that is, \( e_{R+1}^t \), and (iii) \( t > R + q + 1 \) where we have to infer three groups of variables \((e_{R+1-q}^t, e_{t-q}^1, e_t^1)\). The initial case \((t = R + 1)\) is straightforwardly derived as follows:

\[
P (e^t_{R+1} | y^t_1, e^R_1, c^R_1) \propto N (y^t_1; b^t_1 + \beta^T E^R_1 + \beta_0 E^R_3, \sigma^2) \cdot N (E^R_1; \mu^t_{R-1}, S^{-1}_{R-1}) \cdot N (E^R_3; 0, \gamma^2)
\]

(31)

In (23), \( b_1 = \alpha^T y_{t-1} + \eta^T c_t + \rho \) as before, and we use \( \beta_{t-q}^1 = 0 \). The theorem of product of two Gaussians is applied to yield (24) from (23). This forms the initial posterior mean and inverse covariance as follows:

\[
\mu^t_{R-1} = \frac{\beta_0 y^2 (y^t_{R+1} - b^t_{R+1})}{\beta_0^2 y^2 + \sigma^2}, \quad S^{-1}_{R+1} = S^{-1}_{R-1}
\]

(25)

We next deal with the second case; that is, \( R + 1 < t \leq R + q + 1 \). We partition \( e^t_{t-q} \) into three parts: \( E_1 = e^t_{t-q}, E_2 = e^t_{R+1-t}, \) and \( E_3 = e_t \). The parameter vector \( \beta \) for \( e^t_{t-q} \) is accordingly divided into subvectors \( \beta_1 \) (for \( E_1 \)) and \( \beta_2 \) (for \( E_2 \)). We only need to infer \( e^t_{R+1} \); thus \( E_2 \) and \( E_3 \) and the conditional density can be derived as follows:

\[
P (e^t_{R+1} | y^t_1, e^R_1, c^R_1) = P \left( \begin{array}{c}
E_2 \\
E_3
\end{array} \middle| y^t_1, e^R_1, c^R_1 \right)
\]

\[
\propto \mathcal{N} \left( \begin{array}{c}
y_t; b_t + \beta^T E_1 + \beta^T E_2 + \beta_0 E_3, \sigma^2 \end{array} \right) \cdot \mathcal{N} \left( \begin{array}{c}
E_2 \\
E_3
\end{array} ; \mu_{t-1}, S^{-1}_{t-1} \right)
\]

(26)

\[
\mu_{t-1}, S^{-1}_{t-1}, \propto \mathcal{N} (E^t_3; 0, y^2) \]

\[
\sigma \exp \left( -\frac{1}{2} \left( \frac{y^t_1 - b_t}{\sigma^2} \right)^T S_{t-1} (E_2 - \mu_{t-1}) \right)
\]

\[
-\frac{1}{2} \sigma^2 \left( \frac{y^t_1 - b_t}{\sigma^2} - \beta^T E_2 - \beta_0 E_3 \right)^2
\]

(27)

Finally, for the third case \((t > R + q + 1)\), the variables to be inferred \((i.e., e^t_{R+1})\) are partitioned into three groups of variables: \( E_1 = e^t_{t-q}, E_2 = e^t_{R+1-t}, \) and \( E_3 = e_t \). Here, \( E_1 \) and \( E_2 \), when concatenated, yield a vector of the same dimension as \( \mu_{t-1} \), and we partition \( \mu_{t-1} \) accordingly as \( \mu_{t-1}(1) \) and \( \mu_{t-1}(2) \). Similarly, \( S^{-1}_{t-1} \) is partitioned into \((2 \times 2)\) blocks, and we denote them by \( S^{-1}_{t-1} (i, j) \) for \( i, j \in \{1, 2\} \). The posterior can then be written as

\[
P (e^t_{R+1} | y^t_1, e^R_1, c^R_1) = P \left( \begin{array}{c}
E_1 \\
E_2 \\
E_3
\end{array} ; \mu_{t-1}, S^{-1}_{t-1} \right)
\]

\[
\propto \mathcal{N} \left( \begin{array}{c}
y_t; b_t + \beta^T E_2 + \beta_0 E_3, \sigma^2 \end{array} \right) \cdot \mathcal{N} \left( \begin{array}{c}
E_2 \\
E_3
\end{array} \middle| \mu_{t-1}, S^{-1}_{t-1} \right)
\]

(31)
\[ \alpha \exp \left( -\frac{E_2^2}{2\sigma^2} - 1 \frac{1}{2\sigma^2} \left( \frac{y_t - b_t}{\sigma_t} - \beta^T E_2 - \beta_0 E_3 \right)^2 \right) \]

\[ \cdot \sum_{i=1}^{2} \sum_{j=1}^{3} \left( E_{i, j} - \mu_{i, j} \right)^T S_{i, j} \left( E_{i, j} - \mu_{i, j} \right) \]

\[ \alpha \exp \left( -\frac{1}{2} \left( \frac{1}{\gamma^2} + \frac{\beta_0^2}{\sigma^2} \right) E_2^2 - \frac{1}{2} E_1 E_2 + \frac{1}{2} E_2 E_1^T \right) \]

\[ \cdot E_2^T \left( S_{2, 1} (2, 2) + \frac{\beta \beta^T}{\sigma^2} \right) E_2 + E_1 S_{2, 1} (2, 2) E_2 \]

\[ - E_2^T \left( \frac{\beta \beta_0^T}{\sigma^2} \right) E_3 + \left( S_{2, 1} (1, 1) \mu_{i, j} - 1 \right) \]

\[ + S_{2, 1} (1, 2) \mu_{i, j} (2) + \left( S_{2, 1} (2, 2) \mu_{i, j} + 1 \right) \frac{\beta a_t}{\sigma^2} \]

\[ + S_{2, 1} (2, 1) \mu_{i, j} + 1 \frac{\beta a_t}{\sigma^2} E_3 \]

Similar to the second case, we derive \( \mu_t \) and \( S_t \) by rearranging the exponent of (33) as a canonical quadratic form in terms of \( (E_1, E_2, E_3) \). The resulting formulas are as follows:

\[ \mu_t = S_t^{-1} \begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \end{bmatrix}, \]

\[ S_t = \begin{bmatrix} Q_{11} & Q_{12} & Q_{13} \\ Q_{21} & Q_{22} & Q_{23} \\ Q_{31} & Q_{32} & Q_{33} \end{bmatrix}, \]

where

\[ Z_1 = S_{2, 1} (1, 1) \mu_{i, j} + S_{2, 1} (1, 2) \mu_{i, j} (2), \]

\[ Z_2 = S_{2, 1} (2, 2) \mu_{i, j} (2) + S_{2, 1} (2, 1) \mu_{i, j} (1) + \frac{\beta a_t}{\sigma^2}, \]

\[ Z_3 = \frac{\beta_0 a_t}{\sigma^2}, \]

\[ Q_{11} = S_{2, 1} (1, 1), \]

\[ Q_{12} = S_{2, 1} (1, 2), \]

\[ Q_{13} = 0, \]

\[ Q_{22} = S_{2, 1} (2, 2) + \frac{\beta \beta^T}{\sigma^2}, \]

\[ Q_{23} = \frac{\beta \beta_0}{\sigma^2}, \]

\[ Q_{33} = 1 + \frac{\beta_0^2}{\sigma^2}. \]

3. Empirical Study

In this section we empirically test the effectiveness of the proposed sARMA estimation method. In particular we deal with the task of portfolio selection on the real-world dataset comprising daily closing prices from Dow Jones Industrial Average (DJIA).

We consider the task of online portfolio selection (OLPS) problem with real stock return data. We begin with a brief description of the OLPS problem. Assuming there are \( M \) different stocks to invest in daily basis, at the beginning of day \( t \), the historic closing stock prices up to day \( t - 1 \), denoted by \( \{P_{j, t-1} \}_{j=0}^{M-1} \), are available, where \( P_j \) is \( M \)-dim vector whose \( j \)th element \( P_j(j) \) is the price of \( j \)th ticker. Using the information, you decide the portfolio allocation vector \( x_t \), a nonnegative \( M \)-dim vector that sums to 1 (i.e., \( \sum_{j=1}^{M} x_t(j) = 1 \)). Assuming no short positioning is allowed, \( x_t(j) \) is the proportion of the whole budget to be invested in the \( j \)th stock for \( j = 1, \ldots, M \).

The portfolio strategy is thus a function that maps the historic market information (say \( \{P_{j, t-1} \}_{j=0}^{M-1} \)) to the price prediction \( P_t \). The sARMA-based portfolio strategy can be built by estimating sARMA models, one for each stock ticker \( j \), for the stock log-return data; namely, \( y_t = \log P_t - \log P_{t-1} \) (here, we drop the dependency on \( j \) for simplicity). Then the predicted \( \tilde{y}_t \) can be used to decide the proportion of the budget to be invested in the \( j \)th ticker at time \( t \). A reasonable strategy is to make no investment (i.e., \( x_t(j) = 0 \)) if \( \tilde{y}_t < 0 \), while forcing \( x_t(j) \) to be proportional to \( \tilde{y}_t \) if \( \tilde{y}_t > 0 \).

To evaluate the performance of a portfolio strategy, we use the popular (running) relative cumulative wealth (RCW) defined as \( RCW(t) = A_t / A_0 \), where \( A_t \) is the total budget at time \( t \). Thus \( RCW(t) \) indicates the total budget return at time \( t \) compared to the initial budget, and the portfolio strategy strategy that yields high RCW(t) for many epochs \( t \)'s is regarded as a good strategy. Assuming that there is no transaction cost, it is not difficult to see that \( RCW(t) = \prod_{t=1}^{T} r_t x_t \), where we define the price relative vector \( r_t = P_t / P_{t-1} \) (division element-wisely).

Hence, in the sARMA model, it is crucial to accurately forecast the returns, and we compare the model estimated by our cost-sensitive loss with the one using traditional ML estimation. For each approach, we estimate \( M \) sARMA models, one for each stock return, and once the predicted returns \( \tilde{y}_t(j) \)'s at \( t \) are obtained, \( x_t(j) \)'s are decided as follows: \( x_t(j) = 0 \) if \( \tilde{y}_t(j) < \epsilon \) and \( x_t(j) = 1/M \) where \( M \) is the number of \( j \)'s with \( \tilde{y}_t(j) \geq \epsilon \). We also contrast them with the fairly standard market portfolio strategy which sets \( x_t(j) \) to be proportional to the total market volume (i.e., the product of the price and the total number of shares) of the ticker \( j \).

We test the above-mentioned three portfolio strategies on the real-world data, the 30 tickers’ daily closing prices from Dow Jones Industrial Average (DJIA).
In this paper we have introduced a novel ARMA model identification method that exploits the asymmetric loss structure for the financial asset return data. The proposed cost function effectively encodes the goodness of matching in directions between true and model predicted asset returns, which is directly related to ultimate profits in the investment. We have provided the subgradient-based optimization using the trust-region approximation, where it has been empirically shown to work well for the portfolio selection problem in a real-world situation.

4. Conclusion

In this paper we have introduced a novel ARMA model identification method that exploits the asymmetric loss structure for the financial asset return data. The proposed cost function effectively encodes the goodness of matching in directions between true and model predicted asset returns, which is directly related to ultimate profits in the investment. We have provided the subgradient-based optimization using the trust-region approximation, where it has been empirically shown to work well for the portfolio selection problem in a real-world situation.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This study is supported by National Research Foundation of Korea (NRF-2013R1A1A1076101).

References


