Akademie věd České republiky
Ústav teorie informace a automatizace, v.v.i.

Academy of Sciences of the Czech Republic
Institute of Information Theory and Automation

RESEARCH REPORT

Jan Šindelář, Václav Kozmík

# External information in prediction of commodity prices

2233 December 2008

ÚTIA AVČR, v.v.i., P.O.Box 18, 182 08 Prague, Czech Republic
Fax: (+420)286890378, http://www.utia.cas.cz,
E-mail: sindys@volny.cz

# Contents

Název práce: Zpřesnění modelu odhadování vývoje cen na komoditních trzích za pomoci přidání nových kanálů
Autor: Václav Kozmík
Katedra (ústav): Katedra pravděpodobnosti a matematické statistiky
Vedoucí bakalářské práce: Mgr. Jan Šindelář
e-mail vedoucího: sindys@volny.cz

Abstrakt: Předložená práce se zabývá odhadem vývoje ceny na trzích s futures. Hlavním cílem této práce je zjistit, na kterých vstupních datech (cena, objem kontraktů, atd.) závisí námi hledaný odhad budoucí ceny. Za předpokladu určitých zjednodušení je problém převeden na matematický model, který je řešitelný za pomocí metod Bayesovského odhadování. Tato práce prezentuje používané metody odhadu ceny a poté se zaměřuje na svůj hlavní cíl, a to je odhad struktury. Použitý algoritmus je popsán matematicky a dále naprogramován v jazyce Matlab. Výsledky algoritmu na konkrétních datech včetně jejich ekonomické interpretace jsou poslední částí této práce.

Klíčová slova: bayesovské odhadování (učení), odhad struktury, finance

Title: External information in prediction of commodity prices
Author: Václav Kozmík
Department: Department of Probability and Mathematical Statistics
Supervisor: Mgr. Jan Šindelář
Supervisor's e-mail address: sindys@volny.cz

Abstract: Present work deals with a problem of price prediction on futures markets. Main goal of this work is to find out on which input channels (price, volume of contracts, etc.) depends the sought future price. We make further simplifications to be able to present the problem as a mathematical one, which can be solved using Bayesian estimation methods. In this work we present the process of price prediction and then concentrate on the main aim, which is structure determination. Applied algorithm is described mathematically and also programmed in Matlab environment. The results of the algorithm on specific data and their economic interpretation are the last part of this work.

Keywords: bayesian estimation (learning), structure determination, finance

# Symbols and Notation

| | |
|---|---|
| $t$ | time index |
| $s_t$ | state at time $t \in \{-1, 0, 1\}$ |
| $a_t$ | action taken at time $t \in \{-2, -1, 0, 1, 2\}$ |
| $y_t$ | price at time $t$ |
| $\Delta_t$ | price difference between time $t$ and $t+1$ |
| $z\,(.)$, $Z\,(.)$ | loss function |
| $f\,(.)$ | probability density function (pdf) |
| $f\,(.\|.)$ | conditional probability density function (cpdf) |
| $E\,(.)$ | mathematical expectation |
| $\nu(.)$ | Belmann function |
| $\boldsymbol{x}_t$ | data vector containing information valid at time $t$ |
| $x_{t,i}$ | value of data channel $i$ at time $t$ |
| $\boldsymbol{\mathcal{P}}_t = (\boldsymbol{x}_1, .., \boldsymbol{x}_t)$ | information about state of the world at time $t$ |
| $\boldsymbol{\theta}_t = (\alpha_{t,i,j}, \beta_{t,i,k}, \gamma_{t,i,l}, \sigma_{t,i})$ | model parameters at time $t$ |
| $\boldsymbol{\Theta}$ | set of all possible parameter values |
| $\boldsymbol{m}_i$ | model structure of data channel $i$ |
| $M_i$ | set of all possible model structures of data channel $i$ |
| $\boldsymbol{V}_t$ | extended information matrix (square) |
| $\nu_t$ | sample counter |
| $\lambda$ | forgetting factor |
| $\boldsymbol{A}^T$ | denotes transposition of matrix $\boldsymbol{A}$ |

# Chapter 1

# Introduction

In this work we will present Bayesian methods used for prediction of financial market prices. In other words, we want to find the relation between the future price of a commodity and the present state of the world. Current state of the world is taken to be described by a set of related market data, such as previous closing prices, volume of traded contracts in a day, number of active contracts in the market and others. In the experiments section of this work we will present our results on real data. Every trading day we can perform three actions: buy a contract, sell a contract or do nothing. To make the problem simpler, let us assume that we can hold a maximum of +1 contract and a minimum of -1 contract at a time. Our goal is to choose the action which maximizes our profit. In order to be able to solve the problem numerically, we present chosen restrictions and approximations throughout the work.

The main problem of this thesis lies in structure determination. Simply said, we have a large set of possible model structures and we want to choose the one which suits our data best. We have chosen the way of Bayesian hypothesis testing, because it has some advantages over the classical approach. One of them is straightforward choice of the best hypothesis from the set of possible ones. The classical approach usually deals with cases of validity of null hypothesis and the expansion to presented case is not easy, especially for large hypotheses sets. The Bayesian approach also allows us to choose our own loss function to influence the testing results. Another advantage may be seen in the fact that no p-values have to be chosen, which could be a hard task especially for our high dimensional problems with large sample size.

As stated in [2], many works dealing with different aspects of the problem of structure determination have already been published. Our goal is not to extend the theory, which is beyond the scope of this thesis, but to apply the presented algorithm to real world data. We improve the existing trading algorithm to use multiple channel sources by incorporating the structure determination, which is the main contribution of this work (considering that this method is not much examined in case of financial time series).

In chapter 2, you can find the basics of underlying theory for Bayesian methods, together with theory and practical adjustements (loss function, approximations) of trading algorithm which is used to compare our practical results. The presented theory is specialized in the problem of structure determination in chapter 3. We present our practical results in chapter 4 and the work is summarized in chapter 5.

# Chapter 2

# Price prediction

We will need following basic theorem in our work:

**Theorem 1** *[Bayes rule] Conditional probability density function $f(\boldsymbol{x}|\boldsymbol{y})$ of random vector $\boldsymbol{X}$ with known $\boldsymbol{Y}$ is equal to:*

$$f\left(\boldsymbol{x}|\boldsymbol{y}\right)=\frac{f\left(\boldsymbol{x}\right)f\left(\boldsymbol{y}|\boldsymbol{x}\right)}{\int_{\mathbb{R}^n}f\left(\boldsymbol{x}\right)f\left(\boldsymbol{y}|\boldsymbol{x}\right)\mathrm{d}\boldsymbol{x}}=\frac{f\left(\boldsymbol{x}\right)f\left(\boldsymbol{y}|\boldsymbol{x}\right)}{f(\boldsymbol{y})}=\frac{f\left(\boldsymbol{x},\boldsymbol{y}\right)}{f\left(\boldsymbol{y}\right)}$$

*for* $\int_{\mathbb{R}^n}f\left(\boldsymbol{x}\right)f\left(\boldsymbol{y}|\boldsymbol{x}\right)\mathrm{d}\boldsymbol{x}\neq 0$.

The proof can be found in [1].

We suppose that we are trading in the market from an initial day, which we call time 1, until now, an n-th trading day, which we call time $n$. We will denote the price at time $t$ as $y_t$ and our position at time $t$ as $s_t\in\{-1,0,1\}$. In the beginning, we start at zero position, that means $s_1=0$. Our profit measured in money from time $t$ to $t+1$ can be written as

$$\tilde{z}\left(\Delta_t,s_{t+1}\right)=s_{t+1}\Delta_t,$$

where $\Delta_t=y_{t+1}-y_t$ is the price difference between time $t$ and $t+1$. Every day we choose our action taken in the market, we will indicate an action taken at time $t$ as $a_t$. Using this notation we can write loss function (negative profit) equivalently:

$$z\left(\Delta_t,s_t,a_t\right)=-\left(s_t+a_t\right)\Delta_t.$$

In order to receive better practical results we had to make some further modifications of the loss function, which we present in section 2.6. Let us denote the sequence of actions taken from time $t$ to time $\bar{t}$ as $a_{t..\bar{t}}$ and the sequence of price differences $\Delta_{t..\bar{t}}$. With the assumption of additivity, the loss function equal to total negative profit at time $t$ is

$$Z\left(\Delta_{1..t},a_{1..t}\right)=\sum_{i=1}^{t}z\left(\Delta_t,s_t,a_t\right),$$

where $s_1 = 0$ and $s_t = \sum_{i=1}^{t-1} a_t$ for $t > 1$. The assumption of additivity is strong and should be abandoned in our future work. For example, if we reach certain loss at some time in sequence of states, we could not continue trading, which means that the sequence is not applicable, even if it would lead to a high profit at the end.

## 2.1 Optimalization

When at certain time $t$, we want to choose best sequence of actions $(a_t, a_{t+1}, ..)$ to minimize the negative gain (maximize the profit). We will write for simplicity

$$z(\Delta_t, a_t) \equiv z(\Delta_t, a_t, s_t),$$

because we always know our state at time $t$ and the future states can be calculated from our actions taken. We will denote our information about the state of the world at time $t$ as $\boldsymbol{\mathcal{P}}_t$ and restrict ourselves to a finite sequence of states (finite horizon) because we want to be able to solve the problem numerically.

**Theorem 2** *[Stochastic dynamic programming for additive loss function] Let the conditions for regular dynamic programming hold and let the loss function be additive. Then, the optimal strategy $a_{t..\bar{t}}$ can be constructed value-wise against the course of time by taking*

$$\nu(\boldsymbol{\mathcal{P}}_t) = \min_{a_t \in A_t} E[z(\Delta_t, a_t) + \nu(\boldsymbol{\mathcal{P}}_{t+1}) | \boldsymbol{\mathcal{P}}_t, a_t],$$

*where $A_t$ are all possible actions at time $t$ (e.g. lead only to state $s_{t+1} \in \{-1, 0, 1\}$ and preserve the known $s_t$), and starting from*

$$\nu(\boldsymbol{\mathcal{P}}_{\bar{t}+1}) = 0.$$

The proof and details can be found in [4].

Under conditions stated in [4], the symbol $E(.)$ could be interpreted as mathematical expectation. We suppose that our capital is little in comparison with the capital traded in the whole market, which means our actions do not have an impact on market price evolution probabilities and following equality holds for conditional probability density functions (cpdfs):

$$f(\Delta_t | \boldsymbol{\mathcal{P}}_t, a_t) = f(\Delta_t | \boldsymbol{\mathcal{P}}_t).$$

Then we can write

$$E[z(\Delta_t, a_t) | \boldsymbol{\mathcal{P}}_t, a_t] = \int_{\mathbb{R}} z(\Delta_t, a_t) f(\Delta_t | \boldsymbol{\mathcal{P}}_t) \, \mathrm{d}\Delta_t.$$

If the probalities $f(\Delta_t | \boldsymbol{\mathcal{P}}_t)$ were known, we could simply evaluate the integral and choose the best actions. In ordinary case we unfortunately do not know the probabilities, but what we try to do is to learn them from the data we obtain. We split the cpdf into two parts: parameterized model and parameter pdf as follows

$$f(\Delta_t|\boldsymbol{\mathcal{P}}_t) = \int_{\boldsymbol{\Theta}} f(\Delta_t|\boldsymbol{\mathcal{P}}_t, \boldsymbol{\theta}_t)\, f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t)\, \mathrm{d}\boldsymbol{\theta}_t,$$

where $\boldsymbol{\Theta}$ denotes the set of all possible parameter values. We will choose the parameterized model $f(\Delta_t|\boldsymbol{\mathcal{P}}_t, \boldsymbol{\theta}_t)$ and the uncertainty will remain only in $f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t)$. When we advance to the next trading day, we need to know $f(\boldsymbol{\theta}_{t+1}|\boldsymbol{\mathcal{P}}_{t+1})$, also called posterior pdf. The task of getting it is divided into two subtasks – data updating and time updating.

## 2.2 Bayesian filtering

When we obtain new data we update the cpdf $f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t)$ to $f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_{t+1})$. We will denote new data vector obtained at time $t+1$ as $\boldsymbol{x}_{t+1}$. The vector can contain information like closing price of a commodity, volume of traded contracts or opening price. With an assumption that we do not have any further information in the beginning, we have $\boldsymbol{\mathcal{P}}_{t+1} = (\boldsymbol{x}_1, .., \boldsymbol{x}_{t+1}) = (\boldsymbol{\mathcal{P}}_t, \boldsymbol{x}_{t+1})$.

**Proposition 3** *[Bayesian filtration - data updating] Under presented conditions:*

$$f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_{t+1}) = \frac{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{\theta}_t)\, f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t)}{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t)},$$

*where*

$$f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t) = \int_{\boldsymbol{\Theta}} f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{\theta}_t)\, f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t)\, \mathrm{d}\boldsymbol{\theta}_t.$$

*Proof.* Multiple application of Bayes rule:

$$f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_{t+1}) = \frac{f(\boldsymbol{\mathcal{P}}_{t+1}|\boldsymbol{\theta}_t)\, f(\boldsymbol{\theta}_t)}{f(\boldsymbol{\mathcal{P}}_{t+1})} = \frac{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{\theta}_t)\, f(\boldsymbol{\mathcal{P}}_t|\boldsymbol{\theta}_t)\, f(\boldsymbol{\theta}_t)}{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t)\, f(\boldsymbol{\mathcal{P}}_t)} =$$

$$= \frac{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{\theta}_t)\, f(\boldsymbol{\theta}_t, \boldsymbol{\mathcal{P}}_t)}{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t)\, f(\boldsymbol{\mathcal{P}}_t)} = \frac{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{\theta}_t)\, f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t)}{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t)},$$

$$f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t) = \int_{\boldsymbol{\Theta}} f(\boldsymbol{x}_{t+1}, \boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t)\, \mathrm{d}\boldsymbol{\theta}_t = \int_{\boldsymbol{\Theta}} f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{\theta}_t)\, f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t)\, \mathrm{d}\boldsymbol{\theta}_t.$$

As we can see this proposition allows us to update the cpdfs by employing new data. What remains is to select the parameterized model and prior pdf $f(\boldsymbol{\theta}_1)$. Such a choice should be based on expert knowledge. However, we suppose we have a lot of data to update the densities, which means that our selection does not have a big influence. We choose normal distribution because it allows precise and simple computations.

**Proposition 4** *[Bayesian filtration - time updating] Under presented conditions:*

$$f\left(\boldsymbol{\theta}_{t+1}|\boldsymbol{\mathcal{P}}_{t+1}\right) = \int_{\boldsymbol{\Theta}} f\left(\boldsymbol{\theta}_{t+1}|\boldsymbol{\mathcal{P}}_{t+1}, \boldsymbol{\theta}_t\right) f\left(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_{t+1}\right) \mathrm{d}\boldsymbol{\theta}_t.$$

*Proof.* Marginalization and application of bayes rule:

$$f\left(\boldsymbol{\theta}_{t+1}|\boldsymbol{\mathcal{P}}_{t+1}\right) = \int_{\boldsymbol{\Theta}} f\left(\boldsymbol{\theta}_{t+1}, \boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_{t+1}\right) \mathrm{d}\boldsymbol{\theta}_t = \int_{\boldsymbol{\Theta}} f\left(\boldsymbol{\theta}_{t+1}|\boldsymbol{\mathcal{P}}_{t+1}, \boldsymbol{\theta}_t\right) f\left(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_{t+1}\right) \mathrm{d}\boldsymbol{\theta}_t.$$

The cpdf $f\left(\boldsymbol{\theta}_{t+1}|\boldsymbol{\mathcal{P}}_{t+1}, \boldsymbol{\theta}_t\right)$ is called parameter evolution model. It is a very important part of our design, but unfortunately we do not know the correct model to be able to write down the cpdf. We deal with this problem by using exponential forgetting, which is described in section 2.5, though we are aware that this could be one of the weakest points in our design.

## 2.3 Parameterized model

The evaluation of presented integrals is too complex if we stay with the general setup. Because of this we rather choose the family of the cpdfs which is friendly to computation. Following definition can be found along with more details in [4].

**Definition 5** *[Exponential family of parameterized models] The parameterized model belongs to the (dynamic) exponential family if it can be written in the form*

$$f\left(\boldsymbol{x}_{t+1}|\boldsymbol{\theta}_t, \boldsymbol{\mathcal{P}}_t\right) = A\left(\boldsymbol{\theta}_t\right) \exp\left(B^T\left(\boldsymbol{\Psi}_t\right) C\left(\boldsymbol{\theta}_t\right)\right),$$

*where $\boldsymbol{\Psi}_t$ is in our case a finite dimensional data vector determined by experience $\boldsymbol{\mathcal{P}}_t$, e. g. $\boldsymbol{\Psi}_t = \left(\boldsymbol{x}_{t-1}, \boldsymbol{x}_t\right)$ for the second order autoregression model,*
*$A\left(.\right)$ is a non-negative scalar function defined on $\boldsymbol{\Theta}$,*
*$B\left(.\right), C\left(.\right)$ are vector functions.*

This model can be a rough approximation for financial time series, but it allows us to convert updating of pdfs into algebraic calculations. Let us consider for this moment that the parameters of the model are time invariant, symbolically $\boldsymbol{\theta} = \boldsymbol{\theta}_t$ for $t > 0$.

**Theorem 6** *[Bayesian estimation in exponential family] Under the conditions mentioned above, if prior pdf $f\left(\boldsymbol{\theta}\right)$ is chosen from exponential family as well, the time evolution of posterior pdf becomes*

$$f\left(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t\right) = \frac{I\left(\boldsymbol{V}_t, \nu_t\right)}{I\left(\boldsymbol{V}_{t-1}, \nu_{t-1}\right)},$$

*with* $\boldsymbol{V}_t = \boldsymbol{V}_{t-1} + B(\boldsymbol{\Psi}_t)$, $\boldsymbol{V}_0 = 0$, $\nu_t = \nu_{t-1} + 1$, $\nu_0 = 0$, *and*

$$I(\boldsymbol{V}, \nu) = \int_{\boldsymbol{\Theta}} A^{\nu}(\boldsymbol{\theta}) \exp\left(\boldsymbol{V}^T C(\boldsymbol{\theta})\right) f(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta},$$

*where* $f(\boldsymbol{\theta})$ *is a prior pdf. The Bayesian parameter estimate (posterior pdf) is*

$$f(\boldsymbol{\theta}|\boldsymbol{\mathcal{P}}_{t+1}) = \frac{A^{\nu_t}(\boldsymbol{\theta}) \exp\left(\boldsymbol{V}_t^T C(\boldsymbol{\theta})\right) f(\boldsymbol{\theta})}{I(\boldsymbol{V}_t, \nu_t)}.$$

The theorem above can be found with explanation and remarks in [4]. Unfortunately, we cannot suppose that the parameters are time invariant, because the relations between data evolve over time. We present an approximative method to deal with this problem in section 2.5.

**Definition 7** *[Gauss-Wishart distribution] The conditional probability density function of future data vector is said to have Gauss-Wishart form, if it can be written as*

$$f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{C}_t, \boldsymbol{W}_t) =$$
$$(2\pi)^{-\boldsymbol{m}_x/2} |\boldsymbol{W}_t|^{1/2} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}_{t+1} - \boldsymbol{C}_t^T \boldsymbol{z}_t\right)^T \boldsymbol{W}_t \left(\boldsymbol{x}_{t+1} - \boldsymbol{C}_t^T \boldsymbol{z}_t\right)\right),$$

*where* $\boldsymbol{m}_x$ *(resp.* $\boldsymbol{m}_z$*) is the dimension of the data vector* $\boldsymbol{x}_{t+1}$ *(resp.* $\boldsymbol{z}_t$*),* $\boldsymbol{z}_t$ *is the regressor, which can be any (also nonlinear) tranformation of past data* $\boldsymbol{\mathcal{P}}_t$*,* $\boldsymbol{C}_t$ *is the regression coefficients* $(\boldsymbol{m}_z, \boldsymbol{m}_x)$ *- dimensional matrix and* $\boldsymbol{W}_t$ *is positive definite* $(\boldsymbol{m}_x, \boldsymbol{m}_x)$ *- matrix of precision. The matrices* $\boldsymbol{W}_t$ *and* $\boldsymbol{C}_t$ *are the unknown parameters of the model.*

In can be seen that the GW distribution is part of the exponential family, see Definition 5. The definition of the GW distribution allows to make some data transformation, like logarithmic transformation, but keeps the computations simple. It can be shown that if the prior distribution has the GW form, then after data updating we obtain distribution which is also in GW form. Moreover if we use exponential forgetting explained in section 2.5, the distribution remains in GW form even after time updating.

## 2.4 Autoregression model

For our computation we suppose that the closing price $y_{t+1}$ at time $t+1$ can be calculated from past data using autoregressive model of second order. From the available information in the market throughout the day, we have chosen the closing price as the one we want to predict. Our information about the state of the world $\boldsymbol{\mathcal{P}}_t = (\boldsymbol{x}_1, .., \boldsymbol{x}_t)$ includes also this price in the past, i. e. one element (also called channel) of the vector $\boldsymbol{x}$ is the closing price. We will denote $n$ the number of elements of vector $\boldsymbol{x}$, meaning we have $n$ data channels. We want to know: $f\left(y_{t+1}|\boldsymbol{\mathcal{P}}_t\right) = f\left(x_{t+1,i}|\boldsymbol{\mathcal{P}}_t\right)$ for some $i \in \{1,..,n\}$. From the marginalization rule we have:

$$f\left(x_{t+1,i}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_t\right) = \int_{\mathbb{R}^{n-1}} f\left(x_{t+1,1},..,x_{t+1,n}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_t\right) \mathrm{d}x_{t+1,1}..\mathrm{d}x_{t+1,i-1}\mathrm{d}x_{t+1,i+1}..\mathrm{d}x_{t+1,n}.$$

And from the Bayes rule:

$$f\left(x_{t+1,1},..,x_{t+1,n}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_t\right) = f\left(x_{t+1,2},..x_{t+1,n}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_t,x_{t+1,1}\right) f\left(x_{t+1,1}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_t\right) = .. =$$

$$= f\left(x_{t+1,n}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_t,x_{t+1,1},..,x_{t+1,n-1}\right) f\left(x_{t+1,n-1}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_t,x_{t+1,1},..,x_{t+1,n-2}\right)..$$
$$..f\left(x_{t+1,1}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_t\right).$$

The model parameters can be split into the separated parameters of each channel, therefore we can write:

$$f\left(x_{t+1,1},..x_{t+1,n}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_t\right) = f\left(x_{t+1,n}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_{t,n},x_{t+1,1},..,x_{t+1,n-1}\right)$$
$$f\left(x_{t+1,n-1}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_{t,n-1},x_{t+1,1},..,x_{t+1,n-2}\right)..f\left(x_{t+1,1}|\boldsymbol{\mathcal{P}}_t,\boldsymbol{\theta}_{t,1}\right).$$

In other words, we can calculate the estimate of the first element of future data vector only from the past data and then recursively calculate all other elements. For our computations, we suppose that all the predicted data could be estimated using autoregression model of second order:

$$x_{t+1,i} = \sum_{j=1}^{n} \alpha_{t,i,j}x_{t-1,j} + \sum_{k=1}^{n} \beta_{t,i,k}x_{t,k} + \sum_{l=1}^{i-1} \gamma_{t,i,l}x_{t+1,l} + \sigma_{t,i}e_{t,i},$$

where $e_{t,i}$ is the noise which is supposed to have normal distribution with 0 mean and dispersion 1. When using this model our vector of unknown parameters $\boldsymbol{\theta}_t$ could be written as

$$\boldsymbol{\theta}_t = \left(\alpha_{t,i,j},\beta_{t,i,k},\gamma_{t,i,l},\sigma_{t,i}\right),$$

where the indices $i,j,k,l$ are all indices of vector $\boldsymbol{x}$. This model can be alternatively rewritten in the form of cpdf which is in Gauss-Wishart form (see Definition 7).

Possibly, data samples can be large (in our terms $n$ could be a seriously big number) and sometimes it is not possible to take all channels into account. We can choose the channels from our expert knowledge or we can use an algorithm for structure determination, whose presentation is the main goal of this work and we will focus on it in the next chapter.

## 2.5 Approximations

### 2.5.1 Time updating

We expect that model parameters evolve in time, but we do not know the rules of their evolution. To obtain at least very simple model of time updating and keep the advantages of simple computations, we have chosen so called *exponential forgetting*. Simply said, we put less significance on older data. We have:

$$f\left(\boldsymbol{\theta}_{t+1}|\boldsymbol{\mathcal{P}}_{t+1}\right) \approx f\left(\boldsymbol{\theta}_{t}|\boldsymbol{\mathcal{P}}_{t+1}\right)^{\lambda},$$

where $\lambda$ is forgetting rate, which is usually set to number close to 1. As result, the only change from the presented Theorem 6 is modified updating of statistics $V$ and counter $\nu$:

$$\boldsymbol{V}_t = \lambda \boldsymbol{V}_{t-1} + B\left(\boldsymbol{\Psi}_t\right),$$

$$\nu_t = \lambda \nu_{t-1} + 1.$$

Above presented solution is clearly not the best one. We hope to improve it in the future at least by using different forgetting factors $\lambda_i$ for each data channel.

### 2.5.2 Parameter estimates

We optimize our action for a horizon of finite length, set to about 15 days of trading. For every day in the time we should (according to the theory) calculate the predicted pdf, but we use learned parameters for whole window up to the horizon and update them when we move the window one day into the future. This reduces significantly computation time and allows us to use longer horizon. When calculating the expected value of certain channel, we should integrate as follows:

$$E\left(x_{t+1,i}|\boldsymbol{\mathcal{P}}_t\right) = \int_{\mathbb{R}} x_{t+1,i} f\left(x_{t+1,i}|\boldsymbol{\mathcal{P}}_t\right) \mathrm{d}x_{t+1,i} =$$

$$= \int_{\mathbb{R}} \int_{\boldsymbol{\Theta}} x_{t+1,i} f\left(x_{t+1,i}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{\theta}_t\right) f\left(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t\right) \mathrm{d}\boldsymbol{\theta}_t \mathrm{d}x_{t+1,i},$$

which takes a lot of computer time. Instead of that, we use the parameter estimates of $(\alpha_{t,i,j}, \beta_{t,i,k}, \gamma_{t,i,l})$, ignore the noise $\sigma_{t,i}e_{t,i}$ and calculate only following summation:

$$E\left(x_{t+1,i}|\boldsymbol{\mathcal{P}}_t\right) = \sum_{j=1}^{n} \hat{\alpha}_{t,i,j} x_{t-1,j} + \sum_{k=1}^{n} \hat{\beta}_{t,i,k} x_{t,k} + \sum_{l=1}^{i-1} \hat{\gamma}_{t,i,l} x_{t+1,l}.$$

We are aware that this is a rough simplification and that we loose part of information gained. On the other hand, if the model and the time variations of the parameters would be selected correctly, the cpdfs for the parameters would concentrate near the mean value.

### 2.5.3 Iterations spread in time

To be able to calculate the best actions in the market, long horizon may be needed, but we are able to optimize the actions only on a relatively short horizon. To correct this we use a procedure called *iterations spread in time.* When we are at the horizon, we want to know our expected loss up to the expiry date of the contract (in case of optimization of futures or option trading) or our expected loss to infinity (in case of stocks), e.g. in notation of Theorem 2 we want to know $\nu\left(\boldsymbol{\mathcal{P}}_{\bar{t}+1}\right)$. This value depends on state $s_{\bar{t}}$ where we end our optimalization. We use as the estimate expected loss from previous iteration (starting from the same state), which means that we need to calculate expected loss for every starting state in each iteration. In our notation we put:

$$\nu\left(\boldsymbol{\mathcal{P}}_{\bar{t}+1}, s_{\bar{t}}\right) = \nu\left(\boldsymbol{\mathcal{P}}_{t-1}, s_{\bar{t}}\right).$$

## 2.6 Loss function

When calculating our loss, we also have to take transaction costs into account, which leads to following modification of loss function:

$$z\left(\Delta_t, s_t, a_t\right) = -\left(s_t + a_t\right)\Delta_t - |a_t|\left(C + S\right),$$

where $C$ stands for commission and $S$ for slippage. We use expert estimates of slippage values (different for each commodity). The sum of commision and slippage is called transaction costs ($T = C + S$).

During our practical experiments we deal with the problem of instability, our algorithm chooses to make many actions in the market, which results in high transaction costs and, moreover, it usually leads to a loss. We want our algorithm to act more strategically, e.g. to choose position $+1$ and hold for some time in case of price increase and the reverse action in case of price decrease. To correct this behaviour we use two penalizations to construct a new loss function. The first one is that we penalize each action even above the transaction costs, denoting new constant $D$ we have:

$$z\left(\Delta_t, s_t, a_t\right) = -\left(s_t + a_t\right)\Delta_t - D|a_t|T.$$

Another technique used is that we penalize the state of being in the market ($|s_t| \neq 0$). This penalization should incorporate the risk which comes from dispersion of the predicted price. Denoting another constant $F$ and $\sigma_t$ estimated dispersion of the predicted price channel in time $t$, we have:

$$z\left(\Delta_t, s_t, a_t\right) = -\left(s_t + a_t\right)\Delta_t - D|a_t|T - F|s_t + a_t|\sigma_t.$$

It can be easily seen that after these changes our loss function remains additive. The practical results of presented modifications can be found in chapter 4. In future we would like to improve the algorithm by using correct utility function which also incorporates the utility of wealth. Abandoning the presumption of additive loss function would obviously lead to higher computational complexity, but we hope that the problem still remains feasible.

# Chapter 3

# Structure determination

Let us assume that, as presented in the previous chapter, the data channels we have can be predicted using autoregression model of second order, which means:

$$x_{t+1,i} = \sum_{j=1}^{n} \alpha_{t,i,j} x_{t-1,j} + \sum_{k=1}^{n} \beta_{t,i,k} x_{t,k} + \sum_{l=1}^{i-1} \gamma_{t,i,l} x_{t+1,l} + \sigma_{t,i} e_{t,i}.$$

What we want to do is to find channels which do really impact the future values of each channel, or generally said, we want to choose the best model available from the set of models $M_i$ for the channel $i$. Let us introduce appropriate flags $a_{i,j}, b_{i,k}, c_{i,l}$ equal to 1 if the corresponding value $(x_{t-1,j}, x_{t,k}, x_{t+1,l})$ influences the predicted one or equal to 0 if not. Then we have that

$$x_{t+1,i} = \sum_{j=1,\, a_{i,j}=1}^{n} \alpha_{t,i,j} x_{t-1,j} + \sum_{k=1,\, b_{i,k}=1}^{n} \beta_{t,i,k} x_{t,k} + \sum_{l=1,\, c_{i,l}=1}^{i-1} \gamma_{t,i,l} x_{t+1,l} + \sigma_{t,i} e_{t,i}.$$

And $M_i$ could be written as

$$M_i = \left\{ \left(a_{i,j}, b_{i,k}, c_{i,l}\right), j = 1..n, k = 1..n, l = 1..i-1, a_{i,j}, b_{i,k}, c_{i,l} \in \{0,1\} \right\}.$$

As you can see we suppose the best model structure to be time invariant. This is a simplification, but it should be noted that solving the prediction with all data available may not be computationally possible. We could handle the model structure uncertainty the same way we deal with model parameters and learn them altoghether, but the calculation of the model structure estimate or integration over all possible models would bring unfeasible complexity.

## 3.1 General approach

At first we will look at the problem more generally. Using Bayesian approach, we could handle the uncertainty about proper model structure $\boldsymbol{m}_i \in M_i$ for the channel $i$ in the same way we do with model parameters. We will choose the prior pdf $f(\boldsymbol{m}_i)$ and when we receive new data, we will update it. We get posterior cpdf as follows:

$$f(\boldsymbol{m}_i|\boldsymbol{\mathcal{P}}_{t+1}) = \frac{f(\boldsymbol{\mathcal{P}}_{t+1}|\boldsymbol{m}_i)\, f(\boldsymbol{m}_i)}{f(\boldsymbol{\mathcal{P}}_{t+1})} = \frac{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i)\, f(\boldsymbol{\mathcal{P}}_t|\boldsymbol{m}_i)\, f(\boldsymbol{m}_i)}{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t)\, f(\boldsymbol{\mathcal{P}}_t)} =$$

$$= \frac{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i)\, f(\boldsymbol{m}_i, \boldsymbol{\mathcal{P}}_t)}{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t)\, f(\boldsymbol{\mathcal{P}}_t)} = \frac{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i)\, f(\boldsymbol{m}_i|\boldsymbol{\mathcal{P}}_t)}{f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t)},$$

$$f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t) = \int_{M_i} f(\boldsymbol{x}_{t+1}, \boldsymbol{m}_i|\boldsymbol{\mathcal{P}}_t)\, \mathrm{d}\boldsymbol{m}_i = \int_{M_i} f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i)\, f(\boldsymbol{m}_i|\boldsymbol{\mathcal{P}}_t)\, \mathrm{d}\boldsymbol{m}_i.$$

The predictive cpdf $f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i)$ could be computed as follows:

$$f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i) = \int_{\boldsymbol{\Theta}} f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i, \boldsymbol{\theta}_t)\, f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i)\, \mathrm{d}\boldsymbol{\theta}_t.$$

This can be done because we can achieve the cpdf $f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i)$ using the techniques decribed in previous chapter.

The article [2] refers that it can be proven, that the posterior cpdf $f(\boldsymbol{m}_i|\boldsymbol{\mathcal{P}}_t)$ concentrates under rather general conditions on a single point. This means that for a large class of loss functions, we could choose the maximum of the cpdf as our structure and we would not loose much generality, although we would reduce the complexity of computations significantly.

The whole algorithm can be now summarized in these steps: In time $t = 0$ choose the prior pdf $f(\boldsymbol{m}_i|\boldsymbol{\mathcal{P}}_0) = f(\boldsymbol{m}_i)$. While data are available do following:

- Evaluate $f(\boldsymbol{x}_{t+1}|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i)$ for all possible structures $\boldsymbol{m}_i$ based on the new data $x_{t+1}$.
- Update $f(\boldsymbol{m}_i|\boldsymbol{\mathcal{P}}_t)$ to $f(\boldsymbol{m}_i|\boldsymbol{\mathcal{P}}_{t+1})$ as shown above.
- Update the time varying parameters $f(\boldsymbol{\theta}_t|\boldsymbol{\mathcal{P}}_t, \boldsymbol{m}_i)$ to $f(\boldsymbol{\theta}_{t+1}|\boldsymbol{\mathcal{P}}_{t+1}, \boldsymbol{m}_i)$ for all possible models using the method od Bayesian filtration explained in the previous chapter.
- Set time $t = t + 1$ and repeat.

The most computational complexity of this algorithm lies in the number of possible structures, e.g. $|M_i| = n_i$ could be a really big number, which complicates storing all the cpdfs and performing integration over $M_i$. Let us imagine that for 30 channels and the model of second order we have at least $2^{60} \approx 10^{18}$ possible structures. It is written in [2] that the model found under the assumption of time invariant parameters contains the best model found without the specified assumption. According to this, we can perform the algorithm in two stages, as written in detail in [2]:

- We find a set of competitive structures under the assumption of time invariant parametes.
- The complete algorithm is applied only to the set of structures found in previous step.

## 3.2 Application to regression-type models

To be able to run the algorithm and receive the results in acceptable time, we have to make some further simplifications. We suppose that the conditional expectation of future values of predicted channels has the Gauss-Wishart form, see Definition 7.

The fact that we restricted ourselves to a special class of hypotheses about the model structure, specifically we only want to know which data channels we can omit, leads to a further simplification - the omitted channel corresponds with zero row of matrix $\boldsymbol{C}_t$ in notation of Definition 7. This allows a further rearrangement and simplification of our computations, details can be found in [2]. Even though we have made many simplifications, the practical task is still not solvable in real time. We had to give up the exact solution of the problem and use the following two-stage algorithm.

In the first stage we suppose the model parameters to be time-invariant. We have a set of available hypotheses $M_i$ and we need to choose the best one of them. We use the maximum likelihood method which reduces the problem to a search for the global maximum of learnt cpdf $f\left(\boldsymbol{m}_i|\boldsymbol{\mathcal{P}}_t\right)$ after we finish processing all available data. This is unfortunately again complicated by a high number of allowed models, because the function is high dimensional and it has been also found to be multi-modal. What we do is that we start with an initial guess of structure and we calculate the probability. Then we calculate the probabilities in a small neighborhood of the initial guess (e.g. with specified set $M_s = \{a_{i,j}, b_{i,k}, c_{i,l}\}$ we get new models in a neighborhood of $M_s$ by changing always only one value in $M_s$) and we compare them to the old one. We take the highest value of all the probabilities and the according model as a new guess and continue until we find a local maximum or repeat too many times (not likely to happen). After there is no new maximum in the neighborhood we make some other initial guess and repeat. If we are to find the same local maximum, we increase the probability that we had found the global one. Usually it is sufficient to start with full and empty regressor as initial guesses, because it is likely that we reach the same regressor starting from both, which greatly increases the probability that we have found the global maximum.

In the second stage we use the structure found in the first stage and we perform the full algorithm with forgetting over the set of all submodels of the found model (e.g. all models that omit more or equal channels than the first one). It should be noted that the lesser forgetting factor we use, the lower number of data channels in regressor should be expected.

The real implementation uses also additional techniques to reduce the computation complexity, using L-D factorization of the information matrix and further tricks. The details can be found with short explanation in [2]. We will discuss the results we obtained on our real world data in the next chapter.

# Chapter 4

# Practical results

We perform our practical experiments on U.S. commodity futures data. We have about 30 data channels which contain daily open, high, low, close prices, volume, commitment of traders data and expert channels supplied by Colosseum a.s. The data contain information from about 4000 trading days. Our testing scripts are programmed in Matlab environment [3]. We use half of the data (about 2000 days) to learn the model structure and parameter estimates and on the second half we start trading the market. We calculate the real gain or loss, which is the main criterion for comparing different settings of our algorithm.

## 4.1 Structure determination

We use the implementation of structure determination algorithm provided by Mr. Kárný (programmed in C). When we use all data channels to fill the maximal regressor, we receive almost the same structure as the maximal one and it takes a long time for the algorithm to finish. This leads us to the opinion that we have to take a smaller subset with channels which we think really do impact the future price.

We have carefully chosen the set of 8 channels, including all price channels (open, high, low, close), the volume and open interest channels, cash price of the contract and one specific expert channel. In this case the results depend mainly on forgetting factor used. When we use forgetting factor set to 1.0, we get almost full regressor for most of the commodities tested. If we set up a forgetting rate, we get for all usable values of forgetting rate (for example for both 0.95 and 0.99) similar models which differ from the full one by having channels *volume* and *open interest* removed. We hope these channels provide useful information for price prediction, but it is not wise to start the learning process with forgetting set to 1.0. The algorithm usually learns the distribution, which changes only slightly with further incoming data thus leading to same decision for every trading day (and being long or short for the whole trading period). This leads us to the

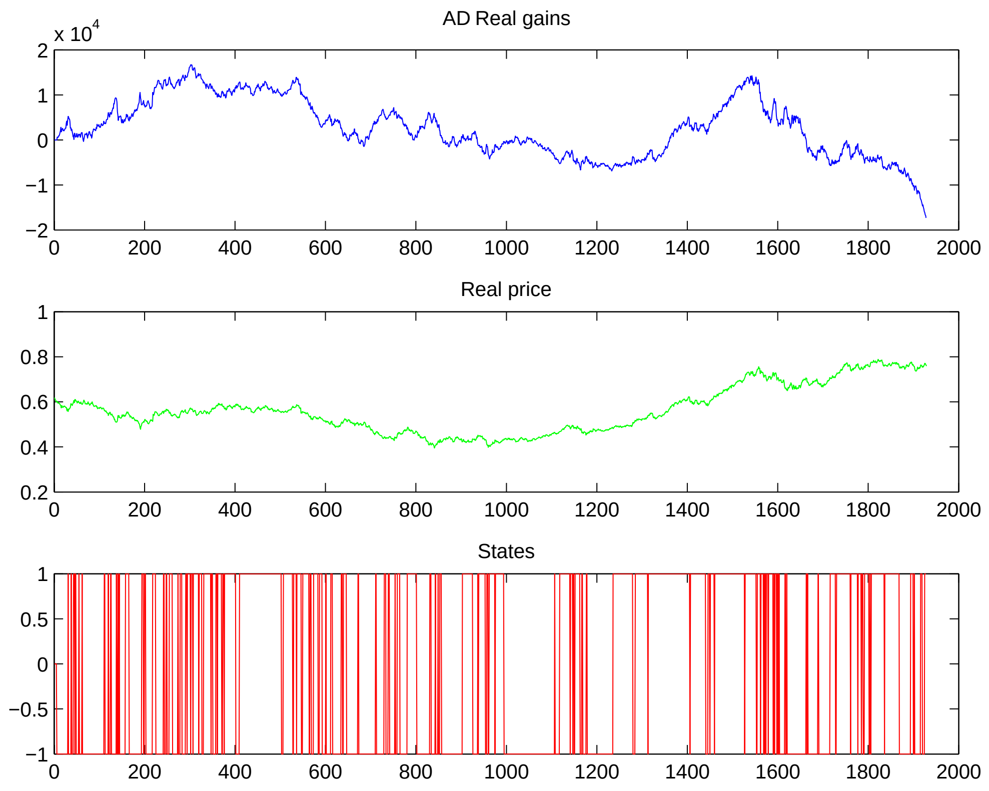

Figure 4.1: Results with no penalization

conclusion that in the future we need to improve our algorithms to be able to handle different forgetting factors for each channel separately.

Another observation made is that the markets with greater price instability tend to return poorer regression structure. Even with no forgetting the channels mentioned previously are removed. Unfortunately, we are not able to give any reasonable economic explanation for this behaviour at the moment. We will present our trading results in the next section.

## 4.2 Action optimalization

To maximize the gain in the market, we suppose that our algorithm should make stable decisions. That means that we would not wish our algorithm to switch to the state $+1$ at time $t$ and change the decision at time $t+3$ to switch to state $-1$. After some practical tests made, we think that to receive stable but accurate price prediction, the ideal forgetting factor is equal approximately to 0.99. As presented on figure 4.1, it can be seen that even with this setup the actions remain unstable. Presented tests are made with the data from Australian Dollar (AD) futures market, horizon set to 15 and using the iterations spread in

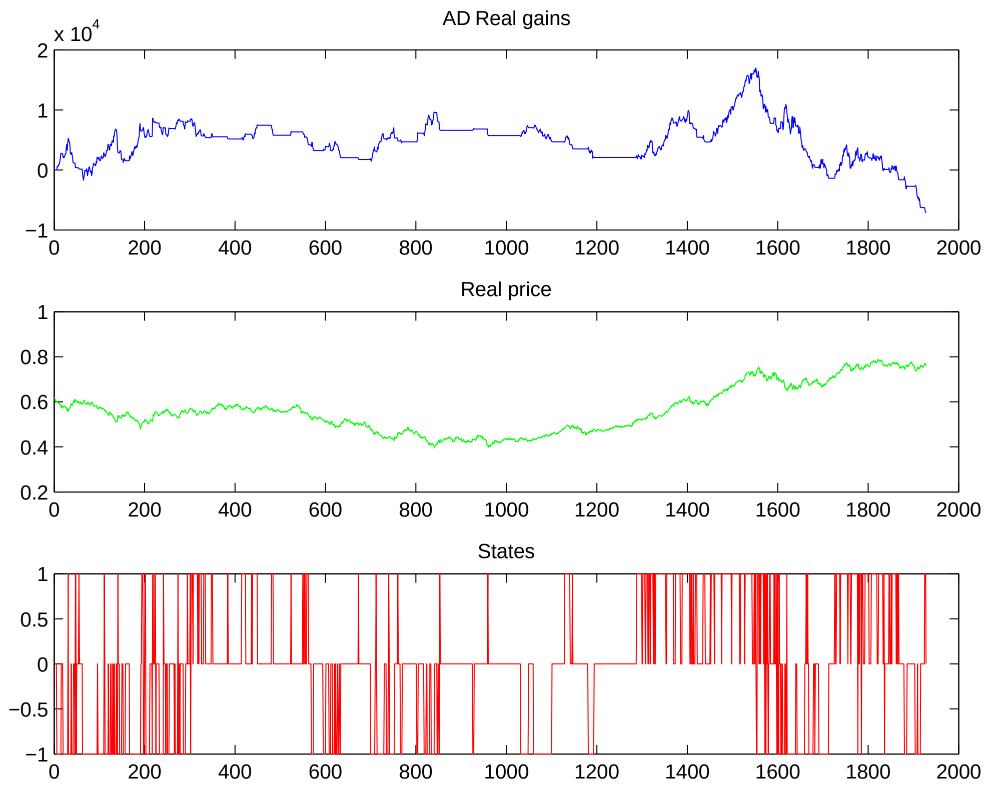

Figure 4.2: Results with state penalization

time method described in section 2.5.

To stabilize the algorithm, we use techniques described in section 2.6. When we set risk constant $(F)$ equal to 500, we receive results in figure 4.2.

These results still do not seem ideal to us. We want to penalize our actions even more and we set the action constant $(D)$ equal to 7 and get results which can be seen in figure 4.3.

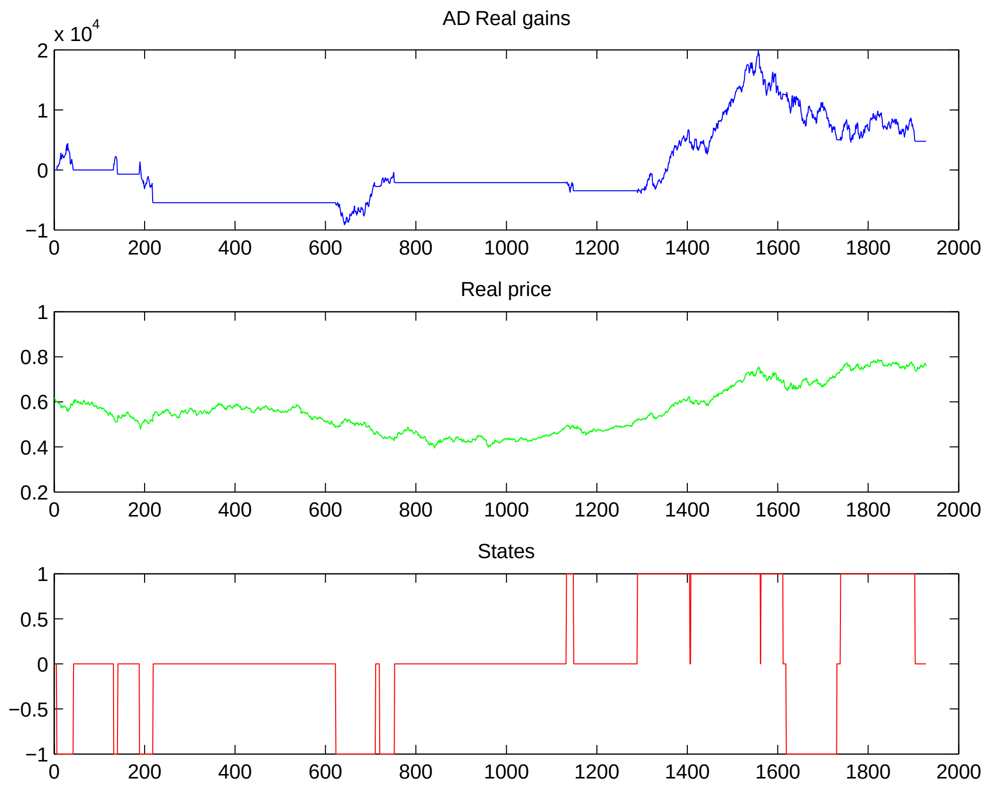

Figure 4.3: Results with state and action penalization

After testing on data from different commodities markets, we can note that the effects of action and state penalization differ for each commodity. Using them both we ended our trading with gain on approximately half of the commodities available. Considering the fact we have made many approximations troughout the optimalization process, we hope that there is still a chance to get better results. In figure 4.4 we present the histogram of total gains reached at the end of trading on 34 commodity markets.

In the last figure 4.5, you can find results with the same parameter settings, but another commodity, this time it is Light Crude Oil (CL).

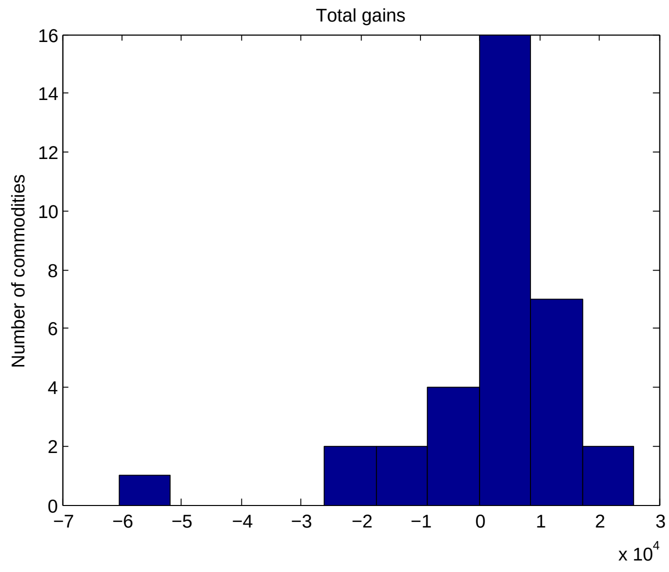

Figure 4.4: Histogram of total gains

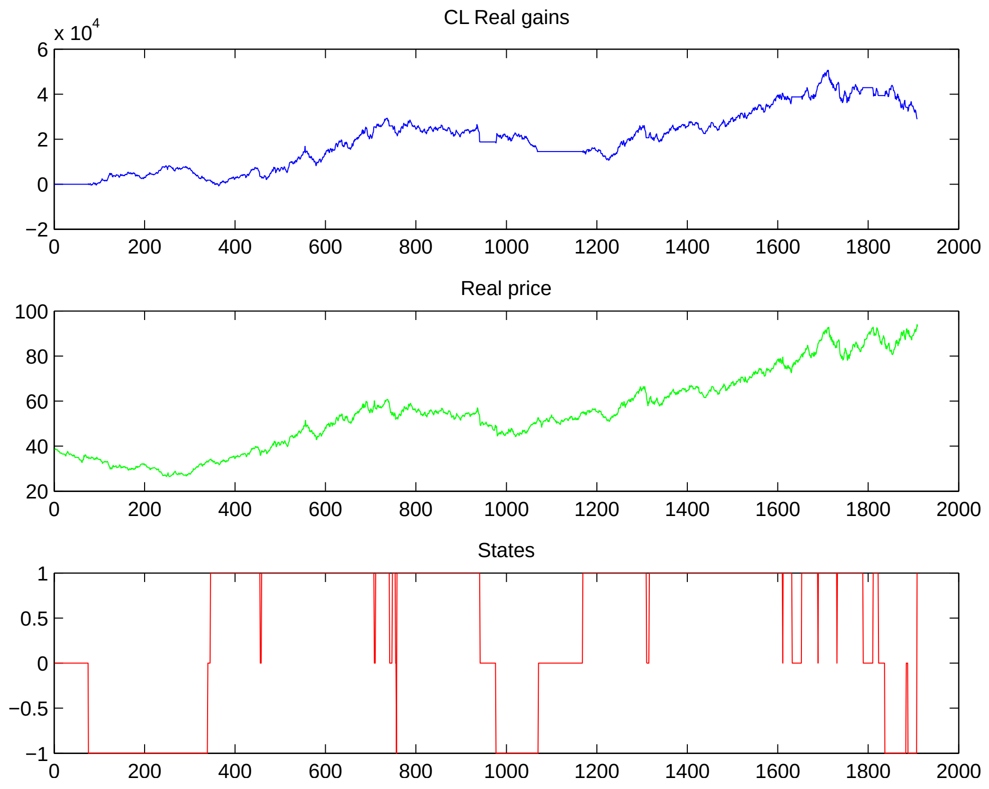

Figure 4.5: Results on another commodity

# Chapter 5

# Conclusion

We managed to implement a new version of optimalization algorithm which includes structure determination and uses multiple channels to predict future price of a commodity. We use exponential forgetting with constant factor and iterations spread in time method to improve our results. The algorithm ends trading with gain on approximately half of the data available, which could be interpreted as approximately staying on its own in general. We are aware that the achieved results are in no means ideal and we want to improve our algorithm in the future. The major improvements could be made by incorporating variable forgetting factor for different channels and markets or by performing an intergration over the predictive cpdf instead of using parameter estimates. Another improvement which could be made is an initial transformation of data, because we expect the prices to have rather log-normal than normal distribution (it is more likely to see price rise of 100 dollars on commodity with actual price of 10000 dollars than on commodity with 100 dollars actual price).

# Bibliography

[1] Anděl, J.: *Základy matematické statistiky*, ISBN 80-7378-001-1, MATFYZPRESS, Praha 2007

[2] Kárný, M. and Kulhavý, R.: *Structure Determination of Regression-Type Models for Adaptive Prediction and Control*, ÚTIA AV ČR, Prague

[3] Matlab: http://www.mathworks.com/products/matlab/

[4] Nagy, I., Pavelková, L., Suzdaleva, E., Homolová, J. and Kárný, M.: *Bayesian Decision Making: Theory and Examples*, ÚTIA AV ČR, Prague 2005

[5] Šindelář, J. and Kárný, M.: *Adaptive Control Applied to Financial Market Data*, Proceedings of WDS 2007, MATFYZPRESS, Prague, 2007