A New Approach to Estimating the Bellman Function

Jan Zeman

***Abstract*— The paper concerns an approximate dynamic programming. It deals with a class of tasks, where the optimal strategy on a shorter horizon is close to the global optimal strategy. This property leads to a new, specific, design of the Bellman function estimation. The paper introduces the proposed approach and provides an illustrative example performed on the futures trading data.**

I. Introduction

The motivation of the research originates from the future trading, with the main aim to design a profitable strategy of buying/selling of commodities, betting on the increase/decrease of the future price [1].

To this aim, the available historical price-data covering 35 markets from the last 15 years has been analyzed. Comparison of the trading strategies designed for different values of time horizon has shown that an increase of amount of data causes only partial change of the strategy designed. Moreover the non-changing part of the strategy is always situated at the beginning and is similar to the best strategy designed for a much larger horizon. This property, specific for futures trading data, has been exploited to design and implement the proposed approach on the approximate dynamic programming.

The paper introduces the mentioned property in more details as well as outlines possible application to dynamic programming.

The dynamic programming is an optimization method based on the idea presented in [2]. The dynamic programming maximizes the gain G over a sequence of decisions $x_t, \dots, x_T$:

$$\max_{x_t,\dots,x_T} G, \tag{1}$$

where $t \in \{1, \dots, T\}$ is a discrete time and T is finite, possibly large, horizon. The set $\{1, \dots, T\}$ is called a decision period.

While dynamic programming searches the argument maximizing $\arg\max_{x_t,\dots,x_T} G$, the maximum the optimal value can be obtained by maximization $\mathcal{V}_t = \max_{x_t,\dots,x_T} G$ is characterized by the Bellman function $\mathcal{V}_t$ [2]. The main drawback of dynamic programming is the curse of dimensionality (see [3]), therefore the approximate solutions should be searched for.

This paper contributes at the approximation of Bellman function. The proposed approach is useful for the tasks arising in economic analysis and trading and can be of interest for other applications.

The Section II-A introduces the dynamic programming and formulates the Bellman equation. The Section II-B deals with the method of a comparison of two strategies, which leads to a design of a system of Bellman equations. The system can be used for an estimation of the Bellman function in a parametric shape (see II-D). The paper is concluded by an example in Section III, where the proposed approach is applied to futures trading data.

II. The Field of Interest

A. Dynamic programming task

A dynamic programming is an method applicable to the problems when it is necessary to find the best decision one after another. The decision making task assumes a *decision maker* and a *system*. The system is a part of the world, which is of interest for the decision maker. The system can be very complex to be fully characterized, moreover the knowledge about the system is usually partial.

The decision maker has own aim related to the system. The aim are expressed in the form of a *gain function* G_τ^T, which quantifies the degree of reaching the aim on (τ, T). The decision maker applies a sequence of decisions $(x_1, \dots, x_T)$ to reach his aims, i.e. maximizing his gain function over the decision period:

$$\max_{x_1,\dots,x_T} G_1^T. \tag{2}$$

The decision maker observes a *system output* $(y_1, \dots, y_T)$. The information available to the decision maker at time t to design a decision x_t is called *knowledge*. The knowledge $\mathcal{P}_t$ contains a history of the system output and previous decisions: $\mathcal{P}_t = (y_1, \dots, y_t, x_1, \dots, x_{t-1})$.

The system and the decision maker form a closed loop. The decision maker enriches his knowledge by system output y_t and designs a decision x_t. The decision can be realized as a system input, which influences the further behavior of the system. This process is repeated at each t up to the horizon T.

At time t, the decision maker maximizes:

$$\max_{x_t,\dots,x_T} G_t^T. \tag{3}$$

The gain function G_t^T depends on the system output over the whole time horizon $(y_t, \dots, y_T)$. However the information available to the decision maker at time t is $\mathcal{P}_t$. Therefore the decision maker is forced to use the *expected value*:

$$\mathcal{E}(a|b) = \int_{a \in a^*} a f(a|b) da,$$

This work was supported by cooperation with Colosseum a.s., CNR-IMATI and grants GACR 102/08/0567, MŠMT 2C06001.

Jan Zeman is with the Department of Adaptive Systems, Institute of Information Theory and Automation, Prague, Czech Republic. zeman@utia.cas.cz

where $\mathcal{E}(a|b)$ is the expected value of the variable a conditioned on the knowledge of variable b and $f(a|b)$ is the probability density function of a defined at the set a^* and conditioned on b.

Thus, the decision maker maximizes the expected value of the gain at time t:

$$\mathcal{V}(\mathcal{P}_t) = \max_{x_t,\ldots,x_T} \mathcal{E}(G_t^T|\mathcal{P}_t, x_t, \ldots, x_T),$$

which defines the Bellman function $\mathcal{V}(\mathcal{P}_t)$.

The assumption of an additive gain function

$$G_{t_1}^{t_2} = G_{t_1}^{t} + G_{t+1}^{t_2} \qquad \text{for } t_1 < t < t_2$$

and the optimality principle [4] allow us to rewrite the Bellman function in the recursive shape:

$$\mathcal{V}(\mathcal{P}_t) = \max_{x_t,\ldots,x_{t+h}} \mathcal{E}(G_t^{t+h} + \mathcal{V}(\mathcal{P}_{t+h+1})|\mathcal{P}_t, x_t, \ldots, x_{t+h})), \tag{4}$$

where the maximum arguments $x_t, \ldots, x_{t+h}$ are the proposed decisions and h is constant, which allows the design of multi-step decision, its value is connected with shape of gain function or kind of task.

The described formulation is too general for the class of tasks considered in futures trading area, therefore the following assumptions are accepted from here onward:

1) Discrete decisions: the decisions are chosen from a finite, discrete and predefined set.

2) Open loop: the decision has no influence on the system.

B. Similarity indexes

For each time t, there is a system output sequence $(y_1, \ldots, y_t)$ available. We design the optimal strategy $X^t = (x_1, \ldots, x_t)$, where we use the time t as horizon. The strategy is optimal only on the time interval $(1, \ldots, t)$, and is denoted by the superscript t.

Designing the strategy X^t at each time t, a sequence of enlarging strategies is obtained:

$$\begin{array}{lcccl}
\{y_1\} & \Rightarrow & \{x_1^1\} & = & X^1, \\
\{y_1, y_2\} & \Rightarrow & \{x_1^2, x_2^2\} & = & X^2, \\
\{y_1, y_2, y_3\} & \Rightarrow & \{x_1^3, x_2^3, x_3^3\} & = & X^3, \\
 & & \vdots & & \\
\{y_1, \ldots\ldots, y_t\} & \Rightarrow & \{x_1^t, \ldots\ldots, x_t^t\} & = & X^t.
\end{array}$$

Let compare the designed strategies with the longest strategy X^T for $t = T$. The strategy X^T is called the optimal strategy, because it is optimal for the decision period , i.e. $\{1, \ldots, T\}$. The other strategies are called suboptimal strategies, because they are not optimal for the whole decision period, but only for the respective sub-periods.

Let us assume that the suboptimal strategies X^t converges to the optimal strategy X^T with the growing t and let take the first t elements of the strategy X^T.

Now we can compare two sequences: $(x_1^T, \ldots, x_t^T)$, which is the beginning part of the optimal strategy X^T and $(x_1^t, \ldots, x_t^t)$, which is the suboptimal strategy designed at time t. To compare these sequences, we used the following similarity indexes:

- *Similarity index S_t :*

$$S_t = \sum_{i=1}^{t} \delta(x_i^t, x_i^T), \tag{5}$$

where $\delta(x, y) = 1$ for $x = y$ and $\delta(x, y) = 0$ for $x \neq y$. The similarity index S_t is a number of identical elements in the sequences $(x_1^T, \ldots, x_t^T)$ and $(x_1^t, \ldots, x_t^t)$.

- *Strict similarity index S_t :*

$$s_t = \max_i \{i; (\forall j \in \mathcal{N})(j \leq i \Rightarrow x_j^t = x_j^T)\}. \tag{6}$$

The strict similarity index is the maximal length of the non-broken identical subsequence beginning by the first element.

The definitions of S_t and s_t imply $s_t \leq S_t \leq t$.

To illustrate the introduces notions, let us consider the following suboptimal and optimal strategies:

$$\begin{array}{llcccccccl}
X^t = & \{\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0\ldots0\ \}, \\
X^T = & \{\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\ldots1\ \},
\end{array}$$

where the sequence X^T is cut to have the same length as X^t. Sequences have 4 elements identical, the fifth element differs, the sixth and seventh elements are identical and then sequences differ.

There, the similarity index $S_t = 6$, because there are 6 identical elements in the sequences. The strict similarity index $s_t = 4$, because the fourth element is the last element, before the first difference occurs.

C. Bellman equation and similarity indexes

The solution of Bellman equation (4) is the most important part of the dynamic programming task. The term 'solution' means finding the Bellman function, a task that can be very complex due to the backward recursive shape of the equation. The optimal actions are only by-products of this solution.

The use of similarity indexes s_t and S_t could is useful, if they grow with the time $s_t \approx t$, $S_t \approx t$.

At each time t, the sequence of optimal actions of length s_t is known and the set of the Bellman equations is:

$$\mathcal{V}(\mathcal{P}_k) = \max_{x_k,\ldots,x_{k+h}} \mathcal{E}(G_k^{k+h} + \mathcal{V}(\mathcal{P}_{k+h+1})|\mathcal{P}_k, x_k, \ldots x_{k+h})), \tag{7}$$

where $k \in \{1, \ldots, s_t - h\}$.

The maximization can be carried out by substitution of suboptimal actions $X^t = (x_1, \ldots, x_{s_t})$:

$$\begin{array}{rcl}
\mathcal{V}(\mathcal{P}_k) & = & \mathcal{E}(G_k^{k+h} + \mathcal{V}(\mathcal{P}_{k+h+1})|\mathcal{P}_k, x_k^t, \ldots, x_{k+h}^t)) \\
 & & \text{for} \quad k \in \{1, \ldots, s_t - h\}.
\end{array} \tag{8}$$

Due to $k \leq t$, the expected values converges to substitution of known values $\mathcal{P}_k$, $(x_k^t, \ldots, x_{k+h}^t)$. Thus, the system of functional equations (8) should be solved to obtain the Bellman function.

D. Parametric shape of Bellman function

A lot of technical details should be resolved before full use of the described approach. We restrict the design to parameterized form of Bellman function:

$$\mathcal{V}(\mathcal{P}_t) \approx V(\mathcal{P}_t;\Theta), \tag{9}$$

where $\Theta \in \Theta^*$ is a vector of unknown parameters. Then, the solution of the Bellman equation converges to estimation of the parameters Θ and data prediction. Inserting (9) into the system of equations (8), one can write:

$$\begin{aligned} V(\mathcal{P}_k;\Theta) + \kappa_k &= \mathcal{E}\big(G_k^{k+h} + V\big(\mathcal{P}_{k+h+1};\Theta\big)| \\ & \qquad \mathcal{P}_k, x_k^t, \ldots, x_{k+h}^t\big), \\ & \qquad \text{for} \quad k \in \{1, \ldots, s_t - h\}. \end{aligned} \tag{10}$$

where κ_k is an error caused by approximation.

The system of functional equations (8) is further reduced to the system of algebraic equation (10).

E. Task classification

Presented design assumes that s_t grows approximately with time t. This is, of course, only the ideal case. Generally there are three types of tasks:

- Task with a strong similarity - is a task, where s_t and S_t grow with the time. Therefore, the number of equations in system (8) or (10) grows with t. Thus, the presented design can be applied.
  In case of use parameterized shape and system (10), it can happen that the number of independent equations overgrows the degree of freedom and the desired solution should be searched respecting that.
- General task without a similarity - where s_t and S_t are small constants independent of t. In this case, the system has a small number of equations. The number of equations in (8) and (10) do not grow, or grow by jumps. There could not be enough equations to find a solution. In this case, different design of the Bellman function should be used. However even the available "poor" system of equations can be used as a prior information about the Bellman function.
- Task with a weak similarity - where s_t is a small constant or growing only by jumps, but S_t grows with t. The proposed approach can be used, but systems (8) and (10) must be written for $k \in \{1, \ldots, S_t - h\}$.
  The approach can be applied carefully not all - but almost all - equations in systems (8) and (10) are valid. Thus the design systematically uses invalid equations and this must be respected.

F. Causality problem

Presented classification is non-causal, because the optimal strategy X^T, designed over all decision period should be known for the calculation of s_t and S_t and the approach can be used for off-line experiments only.

On-line use needs to study the behavior of sequences of the suboptimal strategies $X^1, X^2, \ldots, X^t$ and to estimate the value of s_t.

III. Example: Futures Trading

Futures trading task is a task typically solved by exchange speculators, who know the past price sequence and try to decide, whether to buy or sell an object of interest. A profit is made, when the speculator guesses the direction of the price evolution, otherwise the speculator loses.

A. Futures trading as a game

From out point of view, the futures trading task can be interpreted as turn based game: The player obtains a price y_t at the beginning of each turn $t \in \{1, 2, \ldots, T\}$. He chooses his decision x_t, whether the price should increase $x_t = 1$ or decrease $x_t = -1$, or player can decide not to play for the turn $x_t = 0$. If player changes the choose x_t according to previous decision x_{t-1}, then he pays a transaction cost $C|x_{t-1} - x_t|$. At the beginning of next turn $t+1$, the player makes profit of $(y_{t+1} - y_t)x_t$, therefore when player bets the right way, he makes money, otherwise he loses.

The player tries to maximize his profit up to horizon T:

$$G_1^T = \sum_{t=1}^{T} (y_t - y_{t-1})x_{t-1} - C|x_{t-1} - x_t|.$$

The initial decision is necessary to be defined as $x_0 = 0$.

The described game is a typical optimization problem of dynamic programming (see [4]) and as such it should be solved.

B. Similarity indexes

It is useful to characterize the systems (8) and (10) according the time t, instead of $k \in \{1, \ldots, s_t - h\}$. Thus, we calculate following constants:

$$c_1 = \max_{t \in \{1 \ldots T\}} (t - s_t), \tag{11}$$

$$c_2 = \max_{t \in \{1 \ldots T\}} (t - S_t), \tag{12}$$

and characterize the systems (8) and (10), which is subset of the original set of equations.

The constants c_1, c_2 characterize maximal number of non-optimal decisions in X^t, which is related with the risk of usage the invalid equations in systems of equations (8) and (10). Hence, the less value of c_1, c_2 is better.

The causal estimation of similarity indexes can be done by analyzing differences between the two suboptimal strategies X^{t-1} and X^t , cf. (5) and (6):

$$\begin{aligned} \hat{S}_t &= \sum_{i=1}^{t-1} \delta(x_i^{t-1}, x_i^t), & (13) \\ \hat{s}_t &= \max_i \{i; (\forall j \in \mathcal{N})(j \le i \Rightarrow x_j^{t-1} = x_j^t)\}. & (14) \end{aligned}$$

Analogically can be obtained causal estimation of the constants c_1 and c_2 at the time t:

$$\hat{c}_{1,t} = \max_{i \in \{1, \ldots, t\}} (i - \hat{s}_i), \tag{15}$$

$$\hat{c}_{2,t} = \max_{i \in \{1, \ldots, t\}} (i - \hat{S}_i), \tag{16}$$

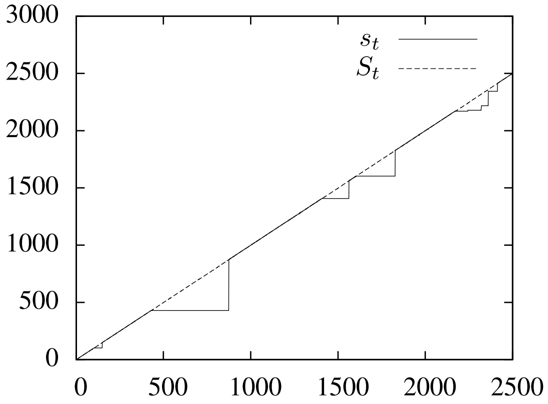

Fig. 1. Example of similarity indexes S_t and s_t for CL

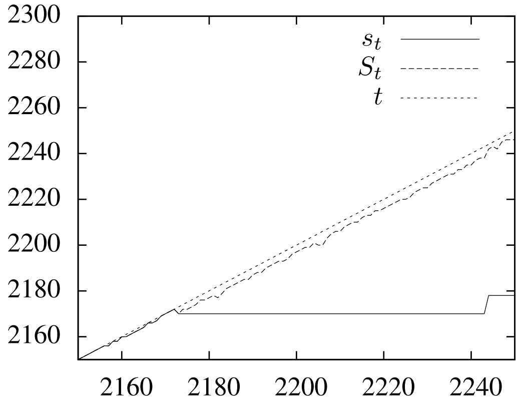

Fig. 2. Example of similarity indexes S_t and s_t for CL (detail)

The final value of $\hat{c}_{1,t}$ and $\hat{c}_{2,t}$ is not so important as their behavior at time $t < T$. The values of $\hat{c}_{1,t}$ and $\hat{c}_{2,t}$ increase with the time t. It is expected that their values converge to a small constant, which is reached very early, therefore the time of the last change $t_{ch;1}$ and $t_{ch;2}$ is documented.

We have 35 price sequences available for the offline experiments. The data were collected once a day, when the exchange was closing, each data set contains data from 1990 to 2005, which makes about 4000 samples all together. Five price sequences were chosen as a representative for the further experiments: Cocoa - CSCE (CC), Petroleum-Crude Oil Light - NMX (CL), 5-Year U.S. Treasury Note - CBT (FV2), Japanese Yen - CME (JY) and Wheat - CBT (W). All constants defined above were estimated for the five reference markets (see Tab. I).

The table shows good results, because the constants c_1 and c_2 are the same and $c_1, c_2 \ll T$. Moreover, four of sequences have c_1 equal to c_2 for each t, which implies that s_t is equal to S_t. The values of $t_{ch;1}$ and $t_{ch;2}$ show the expected fact, that the values of $\hat{c}_{1,t}$ and $\hat{c}_{2,t}$ do not change often and the causal estimation of $\hat{c}_{1,t}$ and $\hat{c}_{2,t}$ gives satisfactory results near to non-causal values. All these facts led to a conclusion that futures trading task is the task with a strong similarity, as was described in Sec. II-E.

The exception with a weak similarity is the market with ticker CL. The obtained similarity indexes are depicted in Fig. 1 and Fig. 2. The difference between s_t and t is markable but it has only a local character, therefore the approach can be used - with the expectation of worse results related to the intervals with a weak similarity.

C. Estimation of Bellman function parameters

Let the parametrized form of Bellman's function be:

$$\mathcal{V}(\mathcal{P}_t) \approx g(x_t)\Psi_t, \tag{17}$$

where $\Psi_t = (y_t, y_{t-1}, \ldots, y_{t-n})^T$ is regressor and $g(x_t)$ is a row vector function.

For illustration purpose, the admissible values for x_t are chosen from a set $x^* = \{-1, 0, 1\}$. Thus, the vector function $g(x_t)$ is fully characterized by $3(n+1)$ parameters, which are the elements of vector Θ introduced in Section II-D. We denote $g(x_t) = (\Theta_{x_t,1}, \Theta_{x_t,2}, \ldots, \Theta_{x_t,n+1})$. Each element $\Theta_{x_t,i}$ is a function of x_t. Due to the chosen set x^*, the function $\Theta_{x_t,i}$ is fully characterized by three values.

Substituting (17) into (10), we obtain:

$$g(x_k^t)\Psi_k - g(x_{k+h+1}^t)\Psi_{k+h+1} = G_k^{k+h} - \kappa_k, \tag{18}$$

for

$$k \in \{1, \ldots, t - c_1 - h\},$$

we get a system of linear equations

$$Ax = b - \mathcal{K} \tag{19}$$

TABLE I
DOMINATING CONSTANTS c_1 AND c_2

Market	c_1	c_2	$\hat{c}_{1,T}$	$\hat{c}_{2,T}$	$t_{ch;1}$	$t_{ch;2}$	T
CC	6	6	7	6	342	342	3822
CL	444	6	446	5	847	2205	3863
FV2	8	8	9	8	383	383	3766
JY	4	4	5	4	50	50	3871
W	7	7	8	7	2452	2452	3822

TABLE II
RESULTS OF EXPERIMENT

Market	MPC	IST
CC	-6 450	-1 490
CL	-12 350	3 390
FV2	-5 701	10 727
JY	-26 568	-35 247
W	-9 792	-1 923

where

$$x = (\Theta_{-1,1}, \dots, \Theta_{-1,n+1}, \Theta_{0,1}, \dots, \Theta_{0,n+1}, \Theta_{1,1}, \dots, \Theta_{1,n+1}).$$

and $\mathcal{K} = (\kappa_1, \kappa_2, \dots, \kappa_{t-c_1-h})$.

The system of linear equations must be solved for each time t to obtain the estimation of the Bellman function values. The number of equations in the system increases by one in each time step. Due to the approximation of the Bellman function, the system need not to be solvable, when the number of equations grows over some threshold. And, an approximate solution of system should be searched. We have applied least square method to minimize the vector of approximation errors $\mathcal{K}$.

D. The results

The obtained parameters are inserted into the parametrized form (17), which is used for maximization of (4). This method corresponds with iterations spread in time (IST) see [5]. To calculate the expected gain, causal predictions generated by autoregressive model were used (see [5]).

As a reference, the results calculated via model predictive control (MPC) were used. The predictive model and task setup were the same for IST.

Final results are summarized in Tab. II. Presented IST method reaches better results than MPC method at four of the five datasets. Neither MPC nor IST gave enough good results satisfactory to the use for real trading. However, the results obtained by IST are slightly better.

IV. Conclusion

The proposed design of the Bellman function is based on searching and analyzing of suboptimal strategies based on known data. The design leads to system of functional equations, but using parametrized shape of Bellman function, the system can be transformed to a system of algebraic equations.

The main idea is to analyze, if the suboptimal strategy contains at least part of the optimal strategy. The task with this property can be either strong or weak similarity. The paper deals with a problem of causal and non-causal analysis leading to a decision which kind of similarity the task exhibits.

The approach is applied and demonstrated on an example of futures trading, which is a typical economic decision making task. The kind of similarity is tested and the behavior of tested method is presented. Then, the new design of Bellman function is applied. Results of experiments are presented and compared to the results of a MPC method and are slightly better.

References

[1] J. Hull, *Options, futures, and other derivatives*. Pearson/Prentice Hall, 2006. [Online]. Available: http://www.worldcat.org/oclc/60321487

[2] R. Bellman, *Dynamic Programming*. Princeton, New Jersey: Princeton University Press, 1957.

[3] W. B. Powell, *Approximate Dynamic Programming*. Wiley-Interscience, 2007.

[4] D. Bertsekas, *Dynamic Programming and Optimal Control*. Nashua, US: Athena Scientific, 2001, 2nd edition.

[5] M. Kárný, B. J., T. V. Guy, L. Jirsa, I. Nagy, P. Nedoma, and L. Tesař, *Optimized Bayesian Dynamic Advising: Theory and Algorithms*. London: Springer, 2005.