A new approach to learning in Dynamic Bayesian Networks (DBNs)

Eric Benhamou, Jamal Atif, Rida Laraki

A new approach to learning in Dynamic Bayesian Networks (DBNs)

Eric Benhamou[*,†] **Jamal Atif**[†] **Rida Laraki**[†]

[*]*A.I. SQUARE CONNECT, 35 Boulevard d'Inkermann 92200 Neuilly sur Seine, France,*
e-mail: eric.benhamou@aisquareconnect.com

†*LAMSADE (UMR CNRS 7243), Université Paris Dauphine, , 75016 Paris, France*
e-mail: eric.benhamou@dauphine.eu; jamal.atif@dauphine.fr; rida.laraki@dauphine.fr

Abstract: In this paper, we revisit the parameter learning problem, namely the estimation of model parameters for Dynamic Bayesian Networks (DBNs). DBNs are directed graphical models of stochastic processes that encompasses and generalize Hidden Markov models (HMMs) and Linear Dynamical Systems (LDSs). Whenever we apply these models to economics and finance, we are forced to make some modeling assumptions about the state dynamics and the graph topology (the DBN structure). These assumptions may be incorrectly specified and contain some additional noise compared to reality. Trying to use a best fit approach through maximum likelihood estimation may miss this point and try to fit at any price these models on data. We present here a new methodology that takes a radical point of view and instead focus on the final efficiency of our model. Parameters are hence estimated in terms of their efficiency rather than their distributional fit to the data. The resulting optimization problem that consists in finding the optimal parameters is a hard problem. We rely on Covariance Matrix Adaptation Evolution Strategy (CMA-ES) method to tackle this issue. We apply this method to the seminal problem of trend detection in financial markets. We see on numerical results that the resulting parameters seem less error prone to over fitting than traditional moving average cross over trend detection and perform better. The method developed here for algorithmic trading is general. It can be applied to other real case applications whenever there is no physical law underlying our DBNs.

Keywords and phrases: DBNs, CMA ES, trend detection, systematic trading.

1. Introduction

As stated in Jordan (2012) (first sentence of the preface), *Graphical models are a marriage between probability theory and graph theory.* They are very powerful as they provide a condensed way to represent variables dependencies. The graphical representation allows not only compacting information. It also provides a powerful formalism for representing and reasoning under uncertainty. It can also represent knowledge about the dynamics of the variables and hence leads to Dynamic Bayesian Networks.

DBNs extend the notion of Bayesian networks as they represents the evolution of the random variables as a function of a discrete sequence, usually a time steps sequence. Hence they are the natural tool to model discrete time chronological observation. As we progress over the sequence, the dynamic terms represented by the model parameters may change while the network architecture stays.The representation as a probabilistic graphical model structured in an acyclic oriented graph enables calculating efficiently conditional probabilities related to model variables.

[*]A.I Square Connect
[†]Lamsade

A typical example of a DBN would be, in medical diagnosis, to determine the probability for a patient to have or host a disease according to his symptoms. This system is made "dynamic" by incorporating the fact that the probability of being sick at time t also depends on past probabilities. Intuitively, this means that risk evolves over time.

1.1. DBN and DAG

More formally, a Bayesian network is a directed acyclic graph (DAG) denoted by $G = (V, E)$, where V is the set of nodes and E the set of edges connecting the nodes. A conditional probability distribution is associated to each node x, and the factorized joint probability on the set of V is given by

$$\mathbb{P}(V) = \prod_{x \in V} \mathbb{P}\big(x \,\big|\, \pi_x\big)$$

where the parents of a node x are denoted by π_x. A dynamic Bayesian Network (DBN) is defined as a pair (B_0, B_{2d}) where B_0 is a traditional Bayesian network representing the initial or *a priori* distribution of random variables, that can be related to time 0 and where B_{2d} is a dynamic two-step Bayesian network describing the transition from time $t-1$ to time t with the probability $\mathbb{P}(x_t \,|\, x_{t-1})$ for any node x belonging to V, in a directed acyclic graph $G = (V, E)$. The joined probability for two sets of nodes V_t and V_{t-1} is given by

$$\mathbb{P}(V_t \,|\, V_{t-1}) = \prod_{x \in V, \pi_x \in V} \mathbb{P}(x_t \,|\, \pi_{x_t}))$$

The factorized joint probability law is computed by *tracing* the sequence in the graph over the time sequence. If we denote by T the total length of the path and by $\mathbf{P}(V_0)$ the joined probability of the initial network B_0, the probability to go from V_0 to T is given by:

$$\mathbb{P}(V_{0:T}) = \mathbb{P}(V_0) \times P(V_{1:T}) = \prod_{x \in V} \mathbb{P}(x_0 \,|\, \pi(x_0)) \times \prod_{t=1}^{T} \prod_{x \in V} \mathbb{P}(x_t \,|\, \pi(x_t))$$

A dynamic Bayesian network thus respects the *Markov property*, which expresses that conditional distributions at time t depend only on the state at time $t-1$. Dynamic Bayesian networks generalize probabilistic models such as Hidden Markov Model (HMM), and Kalman filter (KF). Apart from the mainstream Kalman filter and HMM models whose DAG is given by 1, more complex DBN can include multi input network with connection between observable and previous latent variables as provided by 2. Another example is the combination of Kalman Filtering (KF) model and echo neural network (ESN) as provided by figure 3.

1.2. EM method and its drawback

A majority of DBNs exploit latent variables to make the model more powerful in terms of explanation power. By defining a joint distribution over visible and latent variables, the corresponding distribution of the observed variables is obtained by marginalization. This has the nice property to express relatively complex distributions in terms of more tractable joint distributions over the

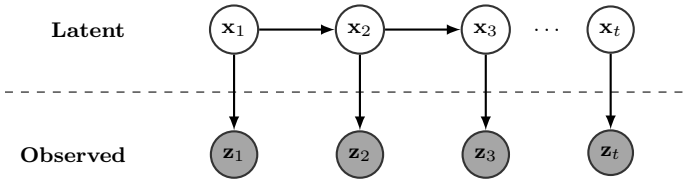

FIG 1. *State Space model as a Bayesian Probabilistic Graphical model. Each vertical slice represents a time step. Nodes in white represent unobservable or latent variables called the states and denoted by* $\mathbf{x}_t$ *while nodes in gray observable ones and are called the spaces and denoted by* $\mathbf{z}_t$*. Each arrow indicates that there is a relationship between the arrow originating node and the arrow targeting node. Dots indicate that there is many time steps. The central dot line is to emphasize the fundamental difference between latent and observed variables. This State Space model encompasses HMM and KF models*

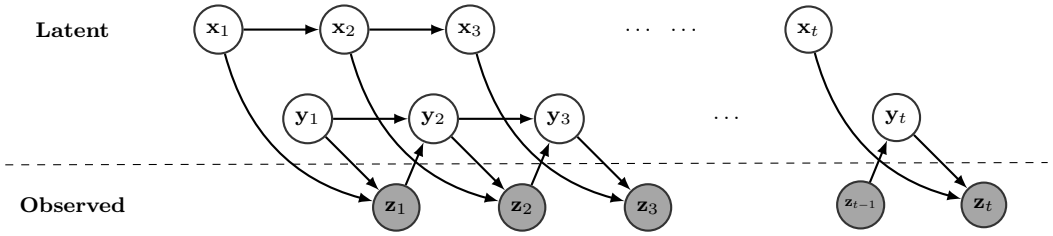

FIG 2. *Example of a dynamic Bayesian network where we have some variables that are observed and some that are latent (non observable). We can see the DAG structure of the network and the Markovian property. This graphical model is more complexed than the traditional state space model as it includes in each time step multi input variables as well as connection between past observable variables and latent variables*

expanded variable space. One well-known example of a hidden variable model is the mixture distribution in which the hidden variable is the discrete component label that provides the corresponding distribution for the observable variable. The static version leads to Gaussian mixture model and more generally the factor analysis model while the dynamic version leads to HMM and in continuous time space to Kalman filter model. However, this does not solve the issue of learning the model parameters. The typical approach is to use the Expectation Maximization (EM) approach to find the gradient coordinate ascent of the Kullback Leibler divergence. The EM algorithm was initially developed for mixture models in particular Gaussian mixtures but also other natural laws from the exponential family such as Poisson, binomial, multinomial and exponential distributions as early as in Hartley (958). It was only once the link between latent variable and Kalman filter models was made that it became obvious that this could also be applied to Kalman and extended Kalman filter (see Cappe et al. (2010) or Einicke et al. (2010)). The EM method is sofar the state of the art method for learning DBNs as it provides an efficient way to find model parameters in a fraction of seconds (see for instance Neal and Hinton (1999), but also Pfeifer and Protzel (2018), Li and McCormick (2017), Robin et al. (2017), Levine (2018)).

However, we argue that although this is a nice method, it misses the point that the DBN is an imprecise model of the reality especially when tackling problem like time series forecasting. In particular, whenever we apply DBNs to economics and finance, we are forced to make some modeling assumptions about the state dynamics and the graph topology (the DBN structure). These assumptions may be incorrectly specified and contain some additional noise compared to

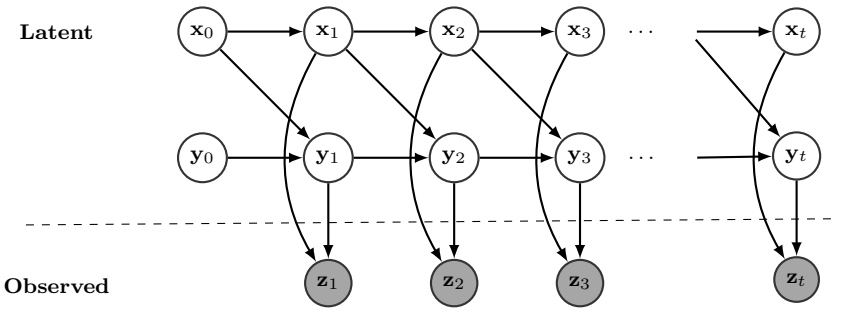

FIG 3. *Example of another dynamic Bayesian network combining Kalman filter (KF) model (and echo neural networks (ESN). This is another example of a multi-input several multi-outputs (MISMO) forecasting model. It is used frequently in time series forecast (see for instance Xiao et al. (2017))*

reality. Trying to use a best fit approach through maximum likelihood estimation and Kullback Leibler divergence optimization may miss this point and try to fit at any price these models on data. We present here a new methodology that takes a radical point of view and instead focus on the final efficiency of our model. Parameters are hence estimated in terms of their efficiency rather than their distributional fit to the data. Our approach relies on Information Geometry optimization and find a local optimum for our final cost function. Our key findings are the following:

- it is possible to directly optimize the cost function with an Evolution Strategies (ES) and in particular the CMA-ES method.
- this approach is a good alternative to the EM approach as it does not fit at all cost the distribution of our network to reality but rather look at model efficiency measured by model cost.
- numerical results shows that the overfitting issue of this approach due to local minimum is less than the EM approach as it incorporates somehow that the model dynamics is incorrectly specified and too simple.

The rest of the paper is organize as follows. Section 2 presents the overall framework and the resulting optimization problem. Section 3 provides some theoretical arguments that favor Evolution Strategies based on Information Geometry Optimization (IGO). Section 4 provides an example in finance of such a method. The method outperforms traditional trend following method by far. We finally conclude about possible extensions of this method and further experiments.

2. Settings

Suppose we have determined an architecture for our network. This may be any of the networks provided by figures 1, 2, 3, or even something different. This model is used for some specific goal. In our case, it will be used to be able to forecast some times series. But this is not our final objective! We are interested in using this forecast to perform a specific action. In the case of a financial markets algorithmic trading strategy, we will use the forecast to make an informed decision and decide whether we should buy or sell a given financial asset. To make things simple, we will assume that when we take our decision, we have in mind a pre-determined strategy. We could imagine a dynamic approach where as time passes we change our objective once we have executed our entry

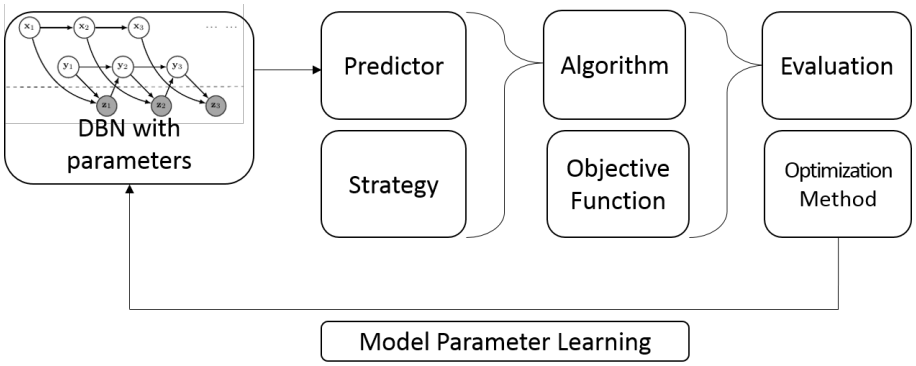

FIG 4. *Learnig process for our DBN. Once an architecture has been decided, we combine this with a strategy to create an systematic algorithm. We select an objective function that provides an Evaluation function. We use an optimization method to find the best parameters according to our evaluation function. We monitor overall performance of the trading strategy on a separate test set to validate scarce overfitting.*

order. For the sake of simplicity, we will assume that whenever we issue an order to trade, we have determined a profit target and a stop loss level for our trade. Although we may have simplified a little bit our setting, this assumption is quite realistic and done by many practitioners as stated in various papers (Labadie and Lehalle (2010), Giuseppe Di Graziano (2014), Fung (2017), or Vezeris et al. (2018)). The profit target ensures that the strategy locks in real money the profit realized and is materialized by a limit order. The stop loss that is physically generated by a stop order safeguards the overall risk by limiting losses whenever the market backfires and contradicts the presumed pattern.

Our final goal at the end of the day is to generate a profitable strategy that does not suffer from overfitting. This is really what matters for us. Hence the EM approach that tackles the overall fit of our model to reality may be inappropriate. This may look quite simple but there are here some real complexity. The overall objective function is often a complex function. In our case, we are interested in maximizing the overall Sharpe ratio of our trading strategy over time and check that the observed performance on the training set does not vanish on the test set. The optimization problem that we are facing is not a simple one as we want to maximize our objective function with respect to our DBN model. The overall goal is summarized in figure 4

2.1. CMA-ES estimation

One of the key point in the selected architecture is the optimization method. In our case, the decision strategy works as follows. If the DBN predicts that the next value of the time series is higher than the last know one plus an offset, we decide to buy the financial asset. Similarly if the DBN predicts that the next value is lower with a sell action. Whenever we issue an order to trade, we also set a profit target and a stop loss in percentage of the last know value. We stay in position until either the trade reaches the profit target and we exit with a profit or it touches the stop loss level and we exit with a loss. The objective function to measure the overall performance of the model is the so called Sharpe ratio. This is a usual performance metric that was established in Sharpe (1966). It is defined in our case as we target absolute performance as trades gains and losses average over the

standard deviation. This number is easy to derive and intuitive to understand as it computes the ratio of the excess return over the strategy standard deviation.

Clearly our objective function is non convex and quite complex to evaluate. It has some discontinuities whenever we reaches critical values. Hence, we rely on CMA-ES algorithm to optimize this evaluation function. CMA ES name stands for covariance matrix adaptation evolution strategy. As it points out, it is an evolution strategy optimization method, meaning that it is a derivative free method that can accommodate non convex optimization problem. The terminology covariance matrix alludes to the fact that the exploration of new points is based on a multinomial distribution whose covariance matrix is progressively determined at each iteration. Hence the covariance matrix adapts in a sense to the sampling space, contracts in dimension that are useless and expands in dimension where natural gradient is steep. This algorithm has led to a large number of papers and articles and we refer to Varelas et al. (2018), Ollivier et al. (2017), Akimoto et al. (2016), Akimoto et al. (2015), Hansen and Auger (2014), Auger and Hansen (2012), Hansen and Auger (2011), Auger and Hansen (2009), Igel et al. (2007), Auger et al. (2004) to cite a few of the numerous articles around CMA-ES. We also refer the reader to the excellent wikipedia page Wikipedia (2018).

CMA-ES relies on two main principles in the exploration of admissible solution for our optimization problem. First, it relies on a multi variate normal distribution as this is the maximum entropy distribution given the first two moments. The mean of the multi variate distribution is updated at each step in order to maximize the likelihood of finding a successful candidate. The second moment, the covariance matrix of the distribution is also updated at each step to increase the likelihood of successful search steps. These updates can be interpreted as a natural gradient descent Ollivier et al. (2017).

Second, we retain two paths of the successive distribution mean, called search or evolution paths. The underlying idea is keep significant information about the correlation between consecutive steps. If consecutive steps are taken in a similar direction, the evolution paths become long. The evolution paths are exploited in two ways. We use the first path is to compute the covariance matrix to increase variance in favorable directions and hence increase convergence speed. The second path is used to control step size and to make consecutive movements of the distribution mean orthogonal in expectation. The goal of this step-size control is to prevent premature convergence yet obtaining fast convergence to a local optimum.

From a practical point of view, we assume that we have a general cost function that depends on our Bayesian graphical model denoted by $\Phi(\theta)$ where θ are the parameters of our Kalman filter. Our cost function is the Sharpe ratio corresponding to a generic trend detection strategy whose signal is generated by our Bayesian graphical model that is underneath a Kalman filter. This approach is more developed in a companion paper Benhamou (2018) but we will give here the general idea. Instead of computing the parameter of our Bayesian graphical model using the EM approach, we would like to find the parameters $\theta_{\max}$ that maximize our cost function $\Phi(\theta)$. Because our cost function is to enter a long trade with a predetermined target level and a given stop loss whenever our Bayesian graphical model anticipates a price risen and similarly to enter a short trade whenever our prediction based on Bayesian graphical model is a downside movement, our trading strategy is not convex neither smooth. It is a full binary function and generates spike whenever there is a trade. Moreover, our final criterium is to use the Sharpe ratio of the resulting trading strategy to compare the efficiency of our parameters. This is way too complicated for traditional optimization method, and we need to rely on Black box optimization techniques like CMA-ES.

3. Properties

3.1. Maximum entropy

One of the theoretical justification of the CMA-ES algorithm is that the multivariate normal distribution for sampling new candidate solutions is the maximum entropy probability distribution over $\mathbb{R}^n$, which translates that it is the sample distribution with the minimal amount of prior information.

3.2. Maximum-likelihood updates

The update for the mean and covariance matrix are built such as to maximize the empirical likelihood. This is in a sense quite similar to the Expectation maximization method as the updates are done first by computing the expectation and then taking the maximum, which leads to

$$m_{k+1} = \arg\max_m \sum_{i=1}^{\mu} w_i \log p_{\mathcal{N}}(x_{i:\lambda}|m) \tag{3.1}$$

where the log-likelihood of x assuming a multivariate normal distribution with mean m and any positive definite covariance matrix C is given by

$$\log p_{\mathcal{N}}(x) = -\frac{1}{2}\log\det(2\pi C) - \frac{1}{2}(x-m)^T C^{-1}(x-m) \tag{3.2}$$

3.3. Natural gradient descent

Last but not least, an interesting feature that provides theoretical justification for the CMA-ES algorithm and that was found by various authors (Akimoto et al. (2010, 2012), Glasmachers et al. (2010) and also the seminal work of Ollivier et al. (2017)) is that the parameters update realizes a natural gradient descent in the space of the sample distributions

4. Numerical Results

In order to test the efficiency of the CMA Es method for Learning parameters in DBNs, we look at the following trend following algorithm based on the following DBN network where we enter a long trade if the prediction of our dynamic Bayesian network forecast is above the close of the previous day and a short trade if the prediction is below the close of the previous day. For each comparison, we add an offset μ to avoid triggering false alarm signals. We set for each trade a pre-determined profit and stop loss target in ticks. These parameters are optimized in order to provide the best sharpe ratio over the train period together with the DBN parameters given by

We take the following HMM model for our DBN (for more details, see Benhamou (2018))

$$\mathbf{x}_{t+1} = \mathbf{\Phi}\mathbf{x}_t + \mathbf{c}_t + \mathbf{w}_t \tag{4.1}$$

$$\mathbf{z}_t = \mathbf{H}\mathbf{x}_t + \mathbf{v}_t \tag{4.2}$$

The noise process $\mathbf{w}_t$ is assumed to follow a multi dimensional normal distribution with zero mean and covariance matrix given by $\mathbf{Q}_t$: $\mathbf{w}_t \sim \mathcal{N}(0, \mathbf{Q}_t)$.

We also assume that the observation noise $\mathbf{v}_t$ follows a multi dimensional normal distribution with zero mean and covariance matrix given by $\mathbf{R}_t$: $\mathbf{v}_t \sim \mathcal{N}(0, \mathbf{R}_t)$. In addition, the initial state, and noise vectors at each step $\mathbf{x}_0, \mathbf{w}_1, \ldots, \mathbf{w}_t, \mathbf{v}_1, \ldots, \mathbf{v}_t$ are assumed to be all mutually independent. We also denote by $\mathbf{P}_t = \mathrm{Cov}(\mathbf{x}_t)$ the covariance matrix of $\mathbf{x}_t$. We assume the following parameters:

$$\mathbf{\Phi}(x) = \begin{bmatrix} p_1 & p_2 \\ 0 & p_3 \end{bmatrix} \quad \mathbf{H} = \begin{bmatrix} p_4 \\ p_5 \end{bmatrix} \quad \mathbf{Q}_{t=0} = \begin{bmatrix} p_6^2 & p_6 p_7 \\ p_7 p_6 & p_8^2 \end{bmatrix} \quad \mathbf{R}_{t=0} = \begin{bmatrix} p_9 \end{bmatrix} \tag{4.3}$$

$$\mathbf{P}_{t=0} = \begin{bmatrix} p_{10} & 0 \\ 0 & p_{11} \end{bmatrix} \quad \mathbf{c}_t = \begin{bmatrix} p_{12}(p_{13} - K_t) \\ p_{14}(p_{15} - K_t) \end{bmatrix} \tag{4.4}$$

The pseudo code of our algorithm is listed below

Algorithm 1 Kalman filter Trend following algorithm

```
Initialize common trade details
SetProfitTarget( target)                                   ▷ fixed profit target in ticks
SetStopLoss( stop_loss )                                   ▷ fixed stop loss in ticks

while Not In Position do                                   ▷ look for new trade
    if DBN( p_1, ..., p_n).Predict[0] ≥ Close[0] + μ then  ▷ up trend signal
        EnterLong()                                        ▷ market order for the open
    else if DBN( p_1, ..., p_n).Predict[0] ≤ Close[0] + μ then  ▷ down trend signal
        EnterShort()                                       ▷ market order for the open
    end if
end while
```

Our resulting algorithm depends on the following parameters $p_1, \ldots, p_n$ the Kalman filter algorithm, the profit target, the stop loss and the signal offset μ. We could estimate the Kalman filter parameters with the EM procedure, then optimize the profit target, the stop loss and the signal offset μ. However, if by any chance the dynamics of the Kalman filter is incorrectly specified, the noise generated by this wrong specification will only be factored in the three parameters: the profit target, the stop loss and the signal offset μ. We prefer to do a combined optimization of all the parameters. We use daily data of the S&P 500 index futures (whose CQG code is *EP*) from 01Jan2017 to 01Jan2018. We train our model on the first 6 months and test it on the next six months. Deliberately, our algorithm is unsophisticated to keep thing simple and concentrate on the parameter estimation method. The overall idea is for a given set of parameter to compute the resulting sharpe ratio over the train period and find the optimal parameters combination. For a model given by equations (4.1) and (4.2) and parameters specified in (4.3) and (4.3), the optimization encompasses 18 parameters: $p_1, \ldots, p_{15}$, the profit target, the stop loss and the signal offset μ, making it non trivial. We use the CMA-ES algorithm to find the optimal solution. In our optimization, we add some penalty condition to force non meaningful Kalman filter parameters to be zero, namely, we add a L1 penalty on this parameters.

Results are given below

TABLE 1
Optimal parameters

Parameters	p_{01}	p_{02}	p_{03}	p_{04}	p_{05}	p_{06}	p_{07}	p_{08}	p_{09}	p_{10}	p_{11}	p_{12}	p_{13}	p_{14}	p_{15}	offset	stop	target
Value	24.8	0	11.8	46.2	77.5	67	100	0	0	0	0	100	0	0	0	5	80	150

TABLE 2
Train test statistics 1/4

Performance	Net Profit	Gross Profit	Gross Loss	# of Trades	# of Contracts	Avg. Trade	Tot. Net Profit (%)	Ann. Net Profit (%)
Train	5,086 €	11,845 €	-6,759 €	15	15	339.05 €	5.09%	10.59%
Test	4,266 €	11,122 €	-6,857 €	15	15	284.38 €	4.27%	8.69%

TABLE 3
Train test statistics 2/4

Performance	Vol	Sharpe Ratio	Trades per Day	Avg. Time in Market	Max. Drawdown	Recovery Factor	Daily Ann. Vol	Monthly Ann. Vol
Train	6.54%	1.62	0.10	8d14h	-2,941 €	3.510	6.54%	5.72%
Test	6.20%	1.40	0.10	8d19h	-1,721 €	4.948	6.20%	5.32%

TABLE 4
Train test statistics 3/4

Performance	Daily Sharpe Ratio	Daily Sortino Ratio	Commission	Percent Profitable	Profit Factor	# of Winning Trades	Avg. Winning Trade	Max. conseq. Winners
Train	1.62	2.35	49 €	46.67%	1.75 €	7	1,692.09 €	3
Test	1.40	2.05	46 €	46.67%	1.62 €	7	1,588.92 €	2

TABLE 5
Train test statistics 4/4

Performance	Largest Winning Trade	# of Losing Trades	Avg. Losing Trade	Max. conseq. Losers	Largest Losing Trade	Avg. Win/Avg. Loss	Avg. Bars in Trade	Time to Recover
Train	1,776.11 €	8	-844.85 €	3	-1,011.82 €	2.00	6.1	77.00 days
Test	1,609.32 €	8	-857.1 €	2	-860.26 €	1.85	6.2	70.00 days

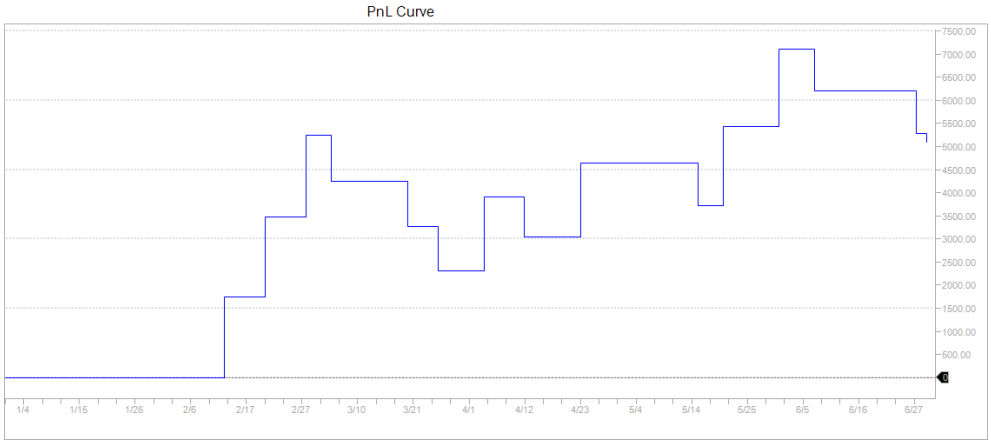

FIG 5. *Kalman filter algorithm on train data set*

We compare our algorithm with a traditional moving average crossover algorithm to test the efficiency of Kalman filter for trend detection. The moving average cross over algorithm generates a buy signal when the fast moving average crosses over the long moving average and a sell signal when the former crosses below the latter. A d period moving average is defined as the arithmetic average of the daily close over a d period, denoted by SMA(d). Our algorithm is given by the following pseudo code

Algorithm 2 Moving Average Trend following algorithm

```
Initialize common trade details
SetProfitTarget( target)                          ▷ fixed profit target in ticks
SetStopLoss( stop_loss )                          ▷ fixed stop loss in ticks

while Not In Position do                          ▷ look for new trade
    if SMA(Short)[0] > SMA(Long)[0] + offset then ▷ up trend signal
        EnterLong()                               ▷ market order for the open
    else if SMA(Short)[0] < SMA(Long)[0] + offset then ▷ down trend signal
        EnterShort()                              ▷ market order for the open
    end if
end while
```

We can now compare moving average cross over versus Kalman filter algorith. The table 6 compares the two algorithms. We can see that on the train period, the two algorithms have similar performances : $5,260$ vs $5,086$. However on the test period, moving average performs very badly with a net profit of 935 versus $4,266$ for the bayesian graphical model (the kalman filter) algorithm.

TABLE 6
Moving average cross over versus Kalman filter

Algo	Total Net Profit	Recovery Factor	Profit Factor	Max. Drawdown	Sharpe Ratio	Total # of Trades	Percent Profitable	Train: Total Net Profit
MA Cross over	935 €	0.32	1.13	-€2,889	0.41	26	0.54	5,260.00
Kalman filter	4,266 €	2.48	1.62	-€1,721	1.40	30	0.47	5,085.79

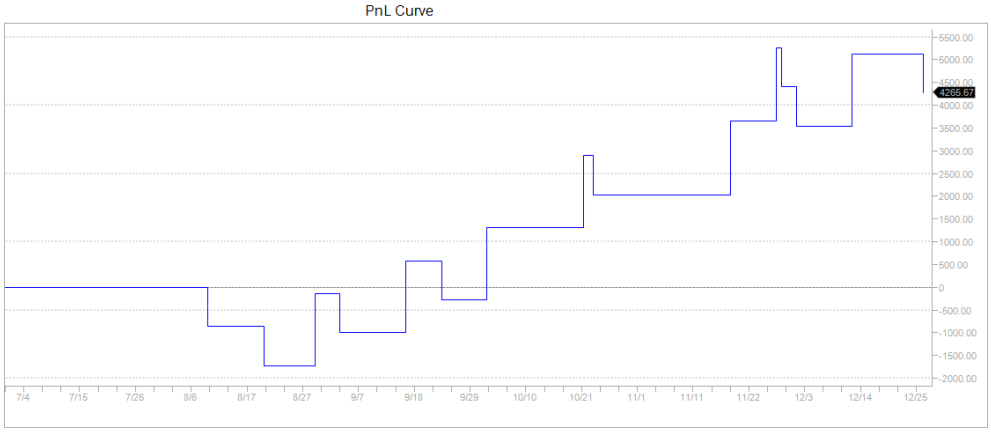

FIG 6. *Kalman filter algorithm on test data set*

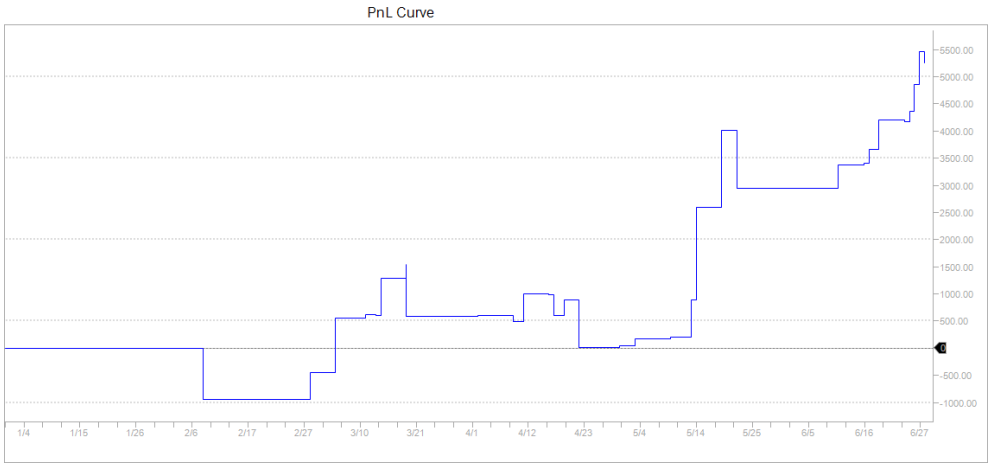

FIG 7. *Moving Average Crossover algorithm on train data set*

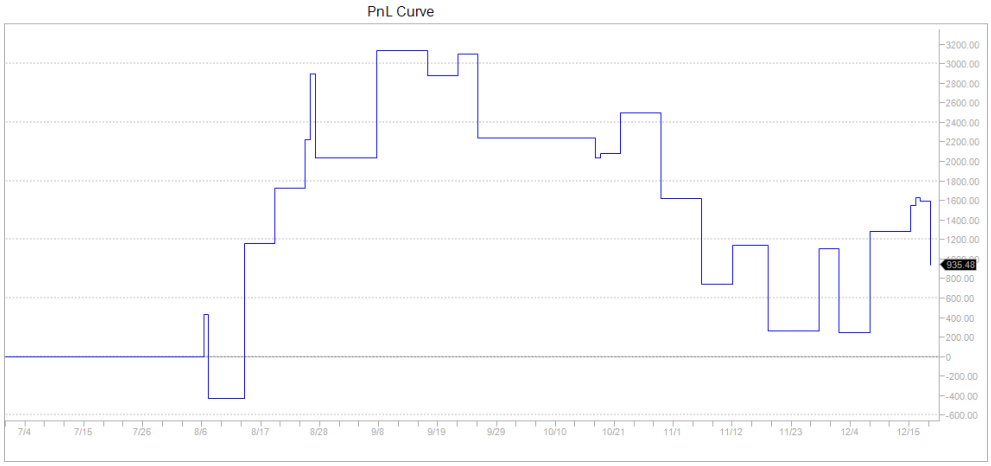

FIG 8. *Moving Average Crossover algorithm on test data set*

5. Conclusion

In this paper, we presented a new method for learning Dynamic Bayesian Networks (DBN) using a new scoring metric that tackles the final usage of our DBN. The main purpose of this work is to present a new method for learning model parameters in DBNs that tackles the final cost function rather than EM that forces the model distribution to fit data at all cost and may result in poor final cost objective function. Thanks to evolutionary optimization techniques, we are able to find local optimum in polynomial time. Using information geometry, we show that the CMA ES method is theoretically sound and robust as it relies on the natural gradient induced by the Fisher information matrix. We conclude than possible extensions are to examine other black box optimization method to check their overall performance and to experiment this approach on other domains.

References

Y. Akimoto, Y. Nagata, I. Ono, and S. Kobayashi. Bidirectional relation between cma evolution strategies and natural evolution strategies. *Lecture Notes in Computer Science - Proceedings of Parallel Problem Solving from Nature - PPSN XI*, 6238:154–163, 2010.

Y. Akimoto, A. Auger, and N. Hansen. Convergence of the continuous time trajectories of isotropic evolution strategies on monotonic c2 -composite functions. In C. A. C. Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, and M. Pavone, editors, *Volume 7491 of Lecture Notes in Computer Science*, pages 2–51. Springer, 2012.

Y. Akimoto, A. Auger, and N. Hansen. Continuous optimization and CMA-ES. In *Genetic and Evolutionary Computation Conference, GECCO 2015, Madrid, Spain, July 11-15, 2015, Companion Material Proceedings*, pages 313–344, 2015.

Y. Akimoto, A. Auger, and N. Hansen. CMA-ES and advanced adaptation mechanisms. In *Genetic and Evolutionary Computation Conference, GECCO 2016, Denver, CO, USA, July 20-24, 2016, Companion Material Proceedings*, pages 533–562, 2016.

A. Auger and N. Hansen. Benchmarking the (1+1)-CMA-ES on the BBOB-2009 noisy testbed. In *Genetic and Evolutionary Computation Conference, GECCO 2009, Proceedings, Montreal, Québec, Canada, July 8-12, 2009, Companion Material*, pages 2467–2472, 2009.

A. Auger and N. Hansen. Tutorial CMA-ES: evolution strategies and covariance matrix adaptation. In *Genetic and Evolutionary Computation Conference, GECCO '12, Philadelphia, PA, USA, July 7-11, 2012, Companion Material Proceedings*, pages 827–848, 2012.

A. Auger, M. Schoenauer, and N. Vanhaecke. LS-CMA-ES: A second-order algorithm for covariance matrix adaptation. In *Parallel Problem Solving from Nature - PPSN VIII, 8th International Conference, Birmingham, UK, September 18-22, 2004, Proceedings*, pages 182–191, 2004.

E. Benhamou. Optimal parameter inference for bayesian graphical models. *ArXiv*, November 2018.

E. Benhamou. Kalman filter demystified: from intuition to probabilistic graphical model to real case in financial markets. *arXiv e-prints*, Nov. 2018.

O. Cappe, E. Moulines, and T. Ryden. *Inference in Hidden Markov Models*. Springer Publishing Company, Incorporated, 2010. ISBN 1441923195, 9781441923196.

G. Einicke, G. Falco, and M. J.T. Em algorithm state matrix estimation for navigation. *IEEE SIGNAL PROCESSING LETTERS*, 17:437–440, 05/2010 2010.

S. P. Y. Fung. Optimal online two-way trading with bounded number of transactions. *CoRR*, 2017.

D. B. A. Giuseppe Di Graziano. Optimal trading stops and algorithmic trading. *SSRN*, 2014. URL https://ssrn.com/abstract=2381830.

T. Glasmachers, T. Schaul, Y. Sun, D. Wierstra, and J. Schmidhuber. Exponential natural evolution strategies. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation GECCO-2010*, pages 393–400. ACM, 2010.

N. Hansen and A. Auger. CMA-ES: evolution strategies and covariance matrix adaptation. In *13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011, Companion Material Proceedings, Dublin, Ireland, July 12-16, 2011*, pages 991–1010, 2011.

N. Hansen and A. Auger. Evolution strategies and CMA-ES (covariance matrix adaptation). In *Genetic and Evolutionary Computation Conference, GECCO '14, Vancouver, BC, Canada, July 12-16, 2014, Companion Material Proceedings*, pages 513–534, 2014.

H. . Hartley. Maximum likelihood estimation from incomplete data. *Biometrics*, 14:174–194, 958.

C. Igel, N. Hansen, and S. Roth. Covariance matrix adaptation for multi-objective optimization. *Evol. Comput.*, 15(1):1–28, Mar. 2007. ISSN 1063-6560.

M. Jordan. *Learning in Graphical Models*. Nato Science Series D:. Springer Netherlands, 2012. ISBN 9789401150149.

M. Labadie and C.-A. Lehalle. Optimal algorithmic trading and market microstructure. Working papers, HAL, 2010.

S. Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *CoRR*, abs/1805.00909, 2018. URL http://arxiv.org/abs/1805.00909.

Z. Li and T. H. McCormick. An Expectation Conditional Maximization approach for Gaussian graphical models. *arXiv e-prints*, Sept. 2017. URL http://arxiv.org/abs/1709.06970.

R. M. Neal and G. E. Hinton. Learning in graphical models. In M. I. Jordan, editor, *Learning in Graphical Models*, chapter A View of the EM Algorithm That Justifies Incremental, Sparse, and Other Variants, pages 355–368. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-60032-3.

Y. Ollivier, L. Arnold, A. Auger, and N. Hansen. Information-geometric optimization algorithms: A unifying picture via invariance principles. *J. Mach. Learn. Res.*, 18(1):564–628, Jan. 2017. ISSN 1532-4435.

T. Pfeifer and P. Protzel. Expectation-maximization for adaptive mixture models in graph optimization. *CoRR*, abs/1811.04748, 2018. URL https://arxiv.org/abs/1811.04748.

G. Robin, C. Ambroise, and S. Robin. Incomplete graphical model inference via latent tree aggregation. *arXiv e-prints*, May 2017. URL http://arxiv.org/abs/1705.09464.

W. F. Sharpe. Mutual fund performance. *Journal of Business*, pages 119–138, January 1966.

K. Varelas, A. Auger, D. Brockhoff, N. Hansen, O. A. ElHara, Y. Semet, R. Kassab, and F. Barbaresco. A comparative study of large-scale variants of CMA-ES. In *Parallel Problem Solving from Nature - PPSN XV - 15th International Conference, Coimbra, Portugal, September 8-12, 2018, Proceedings, Part I*, pages 3–15, 2018.

D. Vezeris, T. Kyrgos, C. T. P. Schinas, and S. Loss. Trading strategies comparison in combination with an macd trading system. *J. Risk Financial Manag*, 11:56, 2018.

Wikipedia. Cma-es, 2018. URL https://en.wikipedia.org/wiki/CMA-ES.

Q. Xiao, C. Chaoqin, and Z. Li. Time series prediction using dynamic bayesian network. *Optik - International Journal for Light and Electron Optics*, 135, 01 2017.