Algorithm for splitting and merging complexes of convex polyhedra according to given hyperplanes in general dimension

Jan Šindelář[1]

[1] Department of Adaptive Systems, Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 4, 18208 Prague, Czech Republic
e-mail:jenik.sindelar@gmail.com

Abstract

In the text algorithm for splitting and merging complexes of convex polyhedra is proposed. The need for such an algorithm emerges for example in computation of normalization factor of Bayesian estimated posterior likelihood of parameters in an auto-regression model with Laplace distributed innovations, where the split of the complex corresponds to newly arrived data incorporation and the merging relates to making such a model adaptive by forgetting older data values – the situation is most apparent in case of an estimation on a moving window.

I. Introduction and motivation

This text was mainly motivated by [1], where the authors introduce an algorithm for effective cutting of complexes of convex polytopes by a hyperplane. As we will see, this algorithm is not well enough suited for our purpose of simultaneously cutting a complex by a hyperplane and merging it according to disappearance of another hyperplane. To arrive at such a situation, we will first introduce a context of modeling time series by an auto-regressive (AR) model and estimation of their parameters in a Bayesian way.

1. Auto-regression model

A class of AR models has been used often for modeling dynamics in various time series. The models are well described in [2]. The AR stochastic process with constant parameters used to model the time series falls into a class of processes linear in the mean value and can be written as

$$\mathbf{\Phi}_{t+1} = \mathbf{A}\mathbf{\Phi}_t + \mathbf{\Sigma}\mathbf{e}_{t+1} \tag{1}$$

where $\mathbf{\Phi}_t$ is a general p dimensional random vector, p is finite, $\mathbf{A}, \mathbf{\Sigma}$ are matrices of unknown parameters and $\mathbf{e}_t$ is the innovation with the properties of white noise with dispersion $\mathbf{I}$ and t is a time index where $t \in \mathcal{T} = \{p, \ldots, T\}$. In case, the dimension of $\mathbf{e}_{t+1}$ is the same as the dimension of $\mathbf{\Phi}_{t+1}$, matrix $\mathbf{\Sigma}$ is a square root of a positive definite matrix and we can reparametrize the model as

$$\mathbf{\Phi}_t = [\mathbf{I} - \mathbf{B}]\,\mathbf{\Phi}_t + \mathbf{B}\mathbf{A}\mathbf{\Phi}_{t-1} + \mathbf{D}^{\frac{1}{2}}\mathbf{e}_t \tag{2}$$

where $\mathbf{B}$ is a lower triangular matrix with units on the diagonal and $\mathbf{D}^{\frac{1}{2}}$ is a diagonal matrix. Thus the vector process can be effectively decoupled into individual AR processes with scalar left-hand side and parameters in these individual scalar models can be estimated separately.

Assuming densities exist for all the random variables involved in modeling the Bayesian estimation of the parameters of the model proceeds with the Bayes rule as new data $\mathbf{d}_t$ arrive

$$f(\boldsymbol{\theta}|\mathbf{d}_t, \mathcal{F}_{t-1}) = \frac{f(\mathbf{d}_t|\boldsymbol{\theta}, \mathcal{F}_{t-1}) f(\boldsymbol{\theta}|\mathcal{F}_{t-1})}{\int_\Omega f(\mathbf{d}_t|\boldsymbol{\theta}, \mathcal{F}_{t-1}) f(\boldsymbol{\theta}|\mathcal{F}_{t-1}) d\boldsymbol{\theta}} \tag{3}$$

where $\boldsymbol{\theta}$ are the parameter values, Ω is the parameter space, $f(\bullet)$ are densities differentiated by their condition and parameter and $\mathcal{F}_t$ is a σ-algebra generated by the information available at time t.

2. *Laplace distributed innovations*

The exact functional form of the estimation depends on the distribution of $\mathbf{e}_t$ in the AR model and on the choice of prior distribution that has to be specified, when only information contained in $\mathcal{F}_0$ is known. If in an individual scalar model, obtained by the above mentioned decomposition

$$Y_t = \boldsymbol{\phi}_t'\boldsymbol{\alpha} + \sigma e_t \tag{4}$$

e_t is Laplace distributed and a conjugate prior is chosen, so that after the estimation, the pre-estimation density of the parameters retains its functional form, the posterior parameter density (likelihood) has the form

$$f(\boldsymbol{\alpha},\sigma|\mathcal{F}_t) = \frac{1}{J_t\sigma^{t+n_1}}\exp\left\{-\frac{1}{\sigma}\sum_{i=1-n_1}^{t}|y_i - \boldsymbol{\phi}_i'\boldsymbol{\alpha}|\right\} \tag{5}$$

where n_0, n_1 are hyperparameters of the conjugate prior and data with indices $i \leq 0$ are artificial data values specified by the choice of the prior as well. J_t is a normalization factor for the density to integrate to 1 over the parameter space.

In case the parameters are not constant in time, but evolve slowly, we can make the model more adaptive by estimating the densities on a moving window of length k only, thus forgetting older data values. Posterior likelihood of the parameters would then read

$$f(\boldsymbol{\alpha},\sigma|\mathcal{F}_t) = \frac{1}{I_t\sigma^{k}}\exp\left\{-\frac{1}{\sigma}\sum_{i=t-k}^{t}|y_i - \boldsymbol{\phi}_i'\boldsymbol{\alpha}|\right\} \tag{6}$$

where I_t is again a normalization factor.

3. *Integrating over parameter space - geometric considerations*

To obtain the normalization factor J_t or I_t or to for example compute moments of the distribution of the parameters, an integral has to be computed over the parameter space Ω. For the purposes of such integration, the parameter space will be split into individual regions

$$\begin{aligned} I_t = \int_\Omega \frac{1}{I_{t-1}\sigma^k}\exp\left[-\frac{1}{\sigma}\sum_{i=t-k}^{t}|y_i - \boldsymbol{\phi}_i'\boldsymbol{\alpha}|\right] d\sigma d\boldsymbol{\alpha} = \\ \int_0^\infty\left(\int_{R_1}\frac{1}{I_{t-1}\sigma^k}\exp\left[-\frac{1}{\sigma}\left(\bar{y}_{t;R_1} - \bar{\boldsymbol{\phi}}_{t;R_1}'\boldsymbol{\alpha}\right)\right]d\boldsymbol{\alpha}\right)d\sigma + \\ + \int_0^\infty\left(\int_{R_2}\frac{1}{I_{t-1}\sigma^k}\exp\left[-\frac{1}{\sigma}\left(\bar{y}_{t;R_2} - \bar{\boldsymbol{\phi}}_{t;R_2}'\boldsymbol{\alpha}\right)\right]d\boldsymbol{\alpha}\right)d\sigma + \cdots \end{aligned}$$

where

$$\begin{gathered} R_i = \left\{\boldsymbol{\alpha} : \forall j \in \mathcal{J}_i, y_j - \boldsymbol{\phi}_j'\boldsymbol{\alpha} \geq 0; \forall l \in \mathcal{L}_i, y_l - \boldsymbol{\phi}_l'\boldsymbol{\alpha} < 0\right\} \\ i \in \{1,2,\dots,N\}\,; \mathcal{J}_i, \mathcal{L}_i \subset \{1,\dots,t\}\,; \mathcal{L}_i = \mathcal{J}_i^C \end{gathered}$$

^{C} stands for complement in the set of indices $\{1,\dots,\tau\}$ and

$$\bar{y}_{t;R_i} = \sum_{j=t-k}^{t}(-1)^{\mathbb{I}(j\in\mathcal{L}_i)}y_j \quad \bar{\phi}_{t;R_i;l} = \sum_{j=t-k}^{t}(-1)^{\mathbb{I}(j\in\mathcal{L}_i)}\phi_{j;l} \tag{7}$$

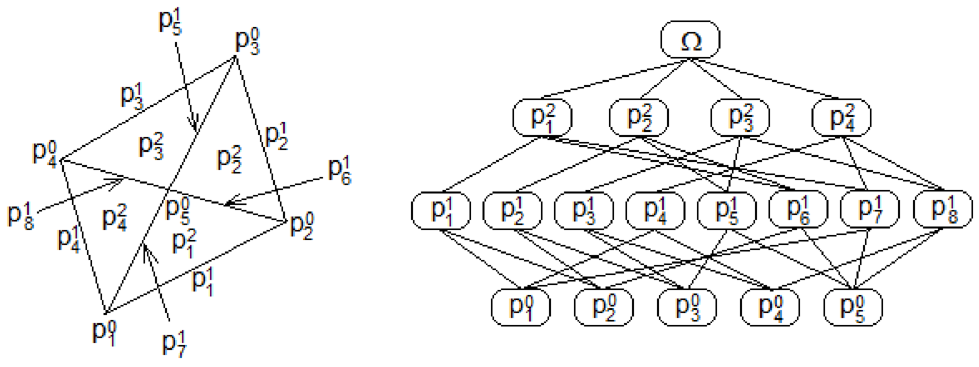
p_5^1
p_3^0
p_3^1
p_4^0
p_3^2
p_2^2
p_2^1
p_6^1
p_8^1
p_4^2
p_5^0
p_2^0
p_4^1
p_1^2
p_1^1
p_1^0
p_7^1
Ω
p_1^2 p_2^2 p_3^2 p_4^2
p_1^1 p_2^1 p_3^1 p_4^1 p_5^1 p_6^1 p_7^1 p_8^1
p_1^0 p_2^0 p_3^0 p_4^0 p_5^0

Figure 1: A Hasse diagram representation of a 2-D single polyhedron.

where index l denotes l-th component of the original vector $\bar{\boldsymbol{\phi}}_{t;R_i}$.

This way the parameter subspace of the location parameters $\boldsymbol{\alpha}$ is divided into a complex of convex polytopes R_i by the hyperplanes $y_i - \boldsymbol{\phi}_i'\boldsymbol{\alpha} = 0$. Some of these regions are infinite in certain direction – we should resolve this property in the following text.

The integrals for each individual polyhedron R_i are computed separately and for the normalization factor they can be exactly solved [3] if coordinates of the vertices are known and if the triangulation of the polyhedron is known. The triangulation can be obtained by a selection procedure theorem described in [4].

The coordinates could be computed by solving the sets of inequalities specifying each individual region serially, but such a computation would be very ineffective. As can be seen, as new data arrive and we move to time $t+1$, the parameter distribution is updated and if the estimation is again restricted to a moving window of length k, one of the hyperplanes vanishes and a new one appears. In this article an effective algorithm will be presented for merging and splitting a complex of convex polytopes.

II. Complex initialization phase

To be able to procede with an effective algorithm for splitting and merging, a suitable representation of the complex at any given time t is needed. In computational geometry two main representations of such a complex are often used.

First is the representation in a form of a binary space partitioning (BSP) tree, used often in computer graphics [5]. Such a representation is not well suited for purposes of high parameter dimensions and especially for purposes of merging.

The second approach is the one used in [1]. Here the complex is represented by a Hasse diagram as shown in the example in Figure 1. A Hasse diagram can be used to represent any partially ordered set. In our case, the Hasse diagram will represent the hierarchy of polyhedrons in an n-dimensional space, where n is the number of location parameters $\boldsymbol{\alpha}$. Each row of the diagram represents all polyhedrons p^m, where m is the dimension, from dimension n in the top row down to dimension 0. This way the points will be in the bottom row, above will be the line segments and so on. The polyhedrons inbetween the rows are connected by an edge, if the lower one is a *facet*, marked $\overset{F}{\subseteq}$, of the top one - the set of vertices of the bottom polyhedron form a subset of the set of vertices of the top polyhedron. In the same way we can represent the whole complex of polyhedrons, with possibly more than one elements in the top row.

The later presented algorithm presents a transformation process of an original Hasse diagram representation into the new one when a hyperplane vanishes and a new one cuts the complex. To procede with the algorithm at least a simple Hasse diagram representation is needed to start with. For this purpose a Hasse diagram representation of the location parameter space with a single point is

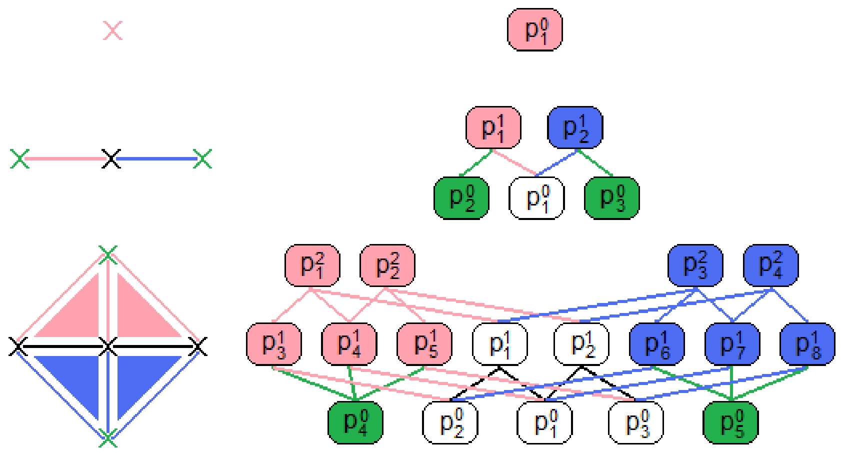

Figure 2: An example of initialization of a space representing Hasse diagram in two dimensions.

constructed in the following way

```
begin
    1. Create a point $p_1^0$ with coordinates $\mathbf{0} = (0, \ldots, 0)$, where $\mathbf{0}$ is $n$-dimensional. Such a point is
       also a simple Hasse diagram $\mathcal{H}_0$.
    2. Set $i = 1$.
    3. Create two copies of Hasse diagram $\mathcal{H}_{i-1}$, move them up one level (points will be segments,
       segments will be 2-D polytopes and so on) and create edges between each offspring and its
       parent.
    4. Create two new *edge points* $p_{2i}^0$ and $p_{2i+1}^0$ with $i$-th coordinate equal to $c_{\max}$ and $-c_{\max}$ respec-
       tively and the other coordinates equal to $0$ and connect each segment of the first copy created in
       previous step to the $p_{2i}^0$ and each segment of the second copy to $p_{2i+1}^0$.
    5. Call the new Hasse diagram $H_i$.
    6. If $i < n$, then $i = i + 1$ and goto step 3.
end
```

The procedure is sketched in Figure 2 in two dimensions. The constant $c_{\max}$ is chosen in respect to physical conditions. An *edge point* is a marked point at the border of the space represented by the created Hasse diagram. This way, we transform an original space, which is infinite into a finite complex of convex polytopes and we remember that behind the edge points, the space continues. This will allow us to proceed with the split/merge algorithm in a simple manner.

An alternative approach to the problem is the use of finite support for parameters $\boldsymbol{\alpha}$. Then we could treat the edge points as regular points defining the parameter space. Such a choice could also be made by the choice of an adequate prior distribution or stabilizing prior distribution. In such a case even the function values on the polytopes could be altered. Such an option will not be further discussed in this paper, but will be left for future research.

The above initialization procedure provides us with a triangulation automatically, since all the polyhedrons in the upper layer of the Hasse diagram are already the basic triangulation polyhedrons in space of dimension n. It is therefore enough to remember the vertices for each polyhedron while building the diagram.

III. Algorithm for splitting and merging the parameter space represented by a Hasse diagram

In this section the main split/merge algorithm should be presented. At time t new data arrive and fix a new condition in the location parameter space

$$\underbrace{y_t - \boldsymbol{\phi}'_t \boldsymbol{\alpha} = 0}_{\spadesuit} \tag{8}$$

this condition splits the parameter space by cutting certain polytopes of current complex represented by a Hasse diagram. At the same time the moving window of length k shifts to the right and forces a single condition

$$\underbrace{y_{t-k-1} - \boldsymbol{\phi}'_{t-k-1} \boldsymbol{\alpha} = 0}_{\clubsuit} \tag{9}$$

to fall out of the list of conditions specifying the estimated density and therefore the Hasse diagram to be merged, where it has been split by $\clubsuit$ before.

As opposed to [1] polyhedrons from the Hasse diagram are allowed to be doubled, when incident conditions result in its multiple recreation. Such an event has a zero probability in many applications, but in a the formulated statistical application and highly discretized real data, omission of such a possibility can lead to high level of unreliability of the model. As well, incident conditions may appear. Such a case will be treated separately, because the Hasse diagram doesn't change, if this happens and a simple linear check through active conditions can save a lot of computational time.

The algorithm proceeds with simultaneous merging, splitting and triangulation altering for polytopes that change

begin

1. Check if the same condition as $\spadesuit$ exists in the list of conditions and if $\clubsuit$ isn't a double condition. If one or the other are true, alter the list accordingly (mark double and single conditions) and apply changes to the top layer of the Hasse diagram (in case of previously presented integration, change the function values for each n-dimensional polytope).

2. $\forall p_i^0$, $i \in \{1, \ldots, N_0\}$, where N_0 is the number of points:

   (a) Classify p_i^0 as $\boxed{+,-,=}_{\spadesuit}$ and $\boxed{+,-,=}_{\clubsuit}$ by plugging the point into related condition equation and remembering the observed sign.

   (b) Send a message $\boxed{+,-,=}_{\spadesuit}$ to all polyhedrons whose vertex p_i^0 is. If it is the last vertex of a given polyhedron mark it $\boxed{\text{S}}$ if it is to be split (it contains $\boxed{+}_{\spadesuit}$ and also $\boxed{-}_{\spadesuit}$ vertex) and add the polyhedron into a list of such polyhedrons (there will be a single list for each Hasse layer). If all vertices are $\boxed{=}_{\spadesuit}$ mark a *single* polyhedron *multi* and mark it $\boxed{=}_{\spadesuit}$.

3. Set $k = 0$.

4. $\forall p_i^k$ classified $\boxed{=}_{\clubsuit}$:

   (a) IF p_i^k is a *single* polyhedron MERGE the parents $\boxed{+}_\clubsuit$ and $\boxed{-}_\clubsuit$ ELSE IF p_i^k is a *multi* polyhedron with multiplicity 2, make it a *single* polyhedron

   (b) Send admissible triangulations to the parent polyhedron and classify admissible triangulation according to the classification theorem in [4].

5. $\forall p_i^k$ classified $\boxed{\text{S}}$:

   (a) Split p_i^k according to [1]. Treat all the $\boxed{=}$ polyhedrons arising from the split as $\boxed{=}_\spadesuit$ (they are equivalent when connecting children and parents).

   (b) After the split obtain all admissible triangulations for all newly created p_h^l from all $p_j^{l-1} \overset{F}{\subseteq} p_h^l$ and classify new admissible triangulations according to [4]. For new points compute their coordinates.

6. IF $k < n$: $k = k + 1$ and goto 4.

end

A MERGE procedure merges two parents of a given $\boxed{=}_\clubsuit$ polyhedron by searching for its $\boxed{+}$ and $\boxed{-}$ parents, merging their edges up and down. After that the $\boxed{=}_\clubsuit$ child is deleted.

According to the purpose of splitting and merging additional operations may be performed on the resultant polyhedrons. The algorithm has to be analyzed to find the most effective way to place them within the algorithm.

IV. Complexity

As can be seen complexity of the algorithm is a critical side to the presented problem. Although not fully resolved yet, partial results are known.

In the initialization phase, we create approximately 3^n polytopes. This phase can be performed offline and therefore shouldn't be problematic.

For the case of the main algorithm, the complexity has not been computed yet. The authors of [1] give an upper complexity bound to the splitting part of the algorithm. Bounds for the number of points linearly classified, maximum number of polytopes marked $\boxed{\text{S}}$ and number of polytopes classified as $\boxed{=}_\clubsuit$ needs to be found and is still an open problem for future research.

V. Acknowledgments

I would like to thank Ministry of Education, Youth and Sports of the Czech Republic for supporting this research through project DAR 1M0572. I would like to thank the Czech Science Foundation for supporting the research through project PUZZLE 102/08/0567. I would also like to thank my supervisor Dr. Miroslav Kárný for his helpful comments on the article.

References

[1] C. Bajaj and V. Pascucci, “Splitting a complex of convex polytopes in any dimension,” in *Proceedings of the twelfth annual symposium on Computational geometry*, pp. 88–97. ACM New York, 1996.

[2] G. Box and G. Jenkins, *Time Series Analysis*, Holden-Day, San Francisco, U.S.A., 1970.

[3] J. Šindelář, “Bayesian vector auto-regression model with laplace errors applied to financial market data,” in *Submitted to MME 2010 conference in České Budějovice, Czech Republic, September 8-10, 2010.*

[4] J. Cohen and T. Hickey, “Two algorithms for detemining volumes of convex polyhedra,” *Journal of the Association for Computing Machinery*, 26:401–414, 1979.

[5] W. Thibault and B. Naylor, “Set operations on polyhedra using binary space partitioning trees,” *Computer Graphics*, 21(4):153–162, 1987.