https://doi.org/10.5194/amt-2020-78
Preprint. Discussion started: 7 April 2020

Mixing height derivation from aerosol lidar using machine learning: KABL and ADABL algorithms

Thomas Rieutord[1], Sylvain Aubert[2], and Tiago Machado[1,2]

[1]Centre National de Recherches Meteorologiques, Université de Toulouse, Météo-France, CNRS, Toulouse, France
[2]Direction des Systèmes d'Observation, Météo-France, Toulouse, France

Correspondence: Thomas RIEUTORD (thomas.rieutord@meteo.fr)

Abstract. Atmospheric boundary layer height (BLH) is a key parameter for several meteorological applications, for example air quality forecast. To measure it, a common practice is to use aerosol lidars: a strong decrease in the backscatter signal indicates the top of the boundary layer. This paper describes and compares two methods of machine learning to derive the BLH from backscatter profiles: the K-means algorithm and the AdaBoost algorithm. Their codes are available under a fully open access, with the name KABL (K-means for Atmospheric Boundary Layer) and ADABL (AdaBoost for Atmospheric Boundary Layer). Both methods are compared to the lidar manufacturer's software and to reference BLH derived from collocated radiosondes. The radiosondes are taken as the reference for all other methods. The comparison is carried out on a two-year long period (2017-2018) on 2 Meteo-France's operational network sites (Trappes and Brest). Results show that, although its training is limited, ADABL is performing better than KABL and can easily be improved by enhancing its training set. However, KABL can be easily adapted for other instrumental device and used to make instrument synergy, while ADABL must be fully re-trained at each change in the instrument settings.

1 Introduction

Atmospheric boundary layer concentrates many scientific challenges (small scale flows, turbulence...) and with high impacts due to its position of interface between ground and atmosphere. For example, air quality forecasts rely on many meteorological parameters, and among them the boundary layer height (BLH) is of first importance. Indeed, the BLH is the depth of atmosphere where all pollutants emitted from the ground will remain. As it varies from a few tenth meters to about 2 km within a day, the dilution/concentration can be very important and responsible for air quality warnings (Stull, 1988; Dupont et al., 2016). Beside, it is one of the largest source of uncertainty in air quality model (Mohan et al., 2011) and there is a need of better evaluation of this parameter (Arciszewska and McClatchey, 2001). In numerical weather prediction models, physical processes are not the same inside the boundary layer (Seity et al., 2011). Therefore it is worth to compare BLH from models and BLH from measurements.

However, measuring the boundary layer height is not straightforward. As stated in Seibert et al. (2000), there is no system matching the requirements to make a reliable estimation of BLH. Best BLH estimation can be achieved through instrument synergy. However, adding instruments limits the eligible sites where the estimation can be made. In this paper we choose to

https://doi.org/10.5194/amt-2020-78
Preprint. Discussion started: 7 April 2020

focus on a single instrument, the aerosol lidar (see section 2.1.1 for more information), already widely used (Haeffelin et al., 2012). The boundary layer is detected by a decrease of the lidar signal at its top. But this decrease can be blurred, perturbed by other strong signals (clouds, residual layers..) and numerical artifacts can occur. For these reasons, there exists numerous studies on BLH derivation from aerosol lidars. Melfi et al. (1985) make a simple thresholding of the signal. Others are based on derivatives: Hayden et al. (1997) take the minimum of the gradient. Menut et al. (1999) use the height of zeroing the second derivative (inflection point) and also the variance of the signal. Senff et al. (1996) use the derivative of the logarithm of the back-scattered intensity along the height. One of the most used method is the wavelet covariance transform (WCT): it looks for the maximum in the convolution between the signal profile and a Haar wavelet (Gamage and Hagelberg, 1993; Cohn and Angevine, 2000; Brooks, 2003). More recent studies are based on backscatter signal analysis, like STRAT (Morille et al., 2007) or CABAM (Kotthaus and Grimmond, 2018). Other studies use graph theory to impose continuity constraints (both vertically and in time) in the BLH estimation: Pathfinder (De Bruine et al., 2017). Inspired by image processing, some use Canny edge detection in addition to backscatter signal analysis (Morille et al., 2007; Haeffelin et al., 2012). STRAT and Pathfinder have been merge into PathfinderTURB by Poltera et al. (2017). All this literature shows that finding BLH from aerosol lidar is still an open question.

Furthermore, artificial intelligence (AI) has reborn in the last decade due to the concomitant increase of available data and computational power. Both reached levels allowing applications that were not possible before. AI has shown some ability to tackle complex classification problems, especially in image classification (Krizhevsky et al., 2012). Such breakthroughs were done thanks to deep convolutional neural networks (LeCun et al., 2015), but AI encompasses much more techniques that also benefit from larger data and computational power (Besse et al., 2018). In this paper we will see how the BLH derivation from backscatter profile can be formulated as a classification problem and we will apply appropriate algorithms to solve it. Toledo et al. (2014) already described a method that falls into the AI scope. It uses unsupervised learning to classify the measure points in or out of the boundary layer. This idea has yielded convincing results in past studies (Toledo et al., 2017; Rieutord et al., 2014). It has been pursued here (under the name of KABL), and explored more carefully. In addition, an alternative machine learning algorithm (named ADABL) was tested. As KABL, it classifies measure points in or out of the boundary layer, but it learns the characteristics of both groups from a training set. To our knowledge, this is the first time that boosting algorithms are applied to this problem.

First, in section 2, we state which data have been used in this study: lidar data in input of algorithms, radiosounding data as reference, ancillary data to sort meteorological conditions. Next, in section 3, we described the two machine learning algorithms (KABL and ADABL) and the procedure to evaluate them. Then, in section 4, we present the results of our study: a sensitivity analysis of the KABL algorithm, a comparison of methods against radiosondes for a 2 years period and case studies. Finally, in section 5, a discussion about the results, the limitations and the prospects of this study is proposed. Last section is dedicated to take-home conclusions.

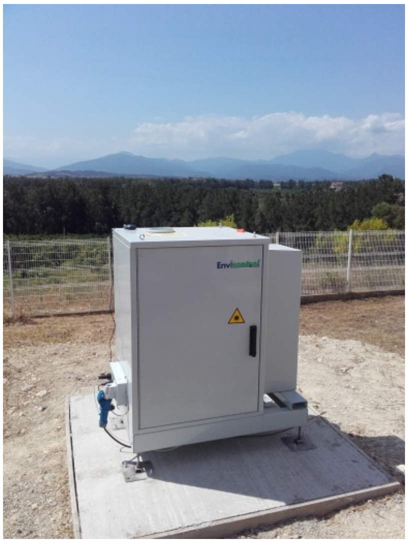

Figure 1. Typical MiniMPL unit from Meteo-France network

2 Material

In this study, the data comes from Meteo-France's operational network. We used collocated radiosoundings and aerosol lidar over two sites : Brest (coastal city in extreme West of France) and Trappes (sub-urban area of Paris, inland). The dataset has a span of two years : 2017 and 2018. A case study is taken on the 2nd of August 2018 in Trappes.

2.1 Lidar data

2.1.1 Lidar network

Starting in 2016, Meteo-France has deployed a network of 6 automatic backscatter lidars to help the Volcanic Ash Advisory Center (VAAC) of Toulouse characterize volcanic ash and aerosols layers. One sensor can be quickly redeployed on a more suitable geographic location depending on the transport event to follow. The network, fully operational since April 2017, is functioning continuously and has been able to detect aerosol events up to an altitude of 17 km. It is part of the wider Automatic Lidars and Ceilometers (ALC) network of the E-PROFILE program described in Haefele et al. (2016).

Two sampling sites of this network have been selected: Brest (48.444 N, 4.412 W, 94 m a.s.l) and Trappes (48.773 N, 2.0124 E, 166 m a.s.l). Each site is equipped with a Mini Micro Pulse LiDAR (MiniMPL), built by Sigma Space Corporation; the exterior casing was provided by Envicontrol. A typical MiniMPL unit from Meteo-France network is shown in figure 1. The MiniMPL is a compact version of the standard MPL systems deployed in the NASA global lidar network (MPLNET). A comprehensive description of the MiniMPL can be found in Ware et al. (2016).

https://doi.org/10.5194/amt-2020-78
Preprint. Discussion started: 7 April 2020

2.1.2 Data processing

The miniMPL acquires profiles of atmospheric backscattering at high frequency (2500 Hz) using a low energy pulse (3.5 μJ) emitted by a Nd:YAG laser at 532 nm. The profiles are acquired in photon-counting mode and, in our present configuration, averaged over 5 minutes and 30 meters vertical resolution bins. The instrument uses a monostatic coaxial design: the laser beam and the receiver optics share the same axis. Due to geometrical limitations, only a fraction of the signal can be recovered in the near field. Therefore the first usable data are provided at 120 meters above ground level on our system.

The instrument has polarization capabilities with the collection of photons on two different channels (more details in Flynn et al. (2007)): the measured raw signals on the "copolarized" and "crosspolarized" channel, respectively suffixed co and cr. These raw signals are then processed to obtain the quantities of interest: the range corrected signal RCS, also called normalized relative backscatter, NRB. This industrial processing comprises several procedures such as background, overlap, afterpulse and dead-time corrections. A comprehensive description of this processing is given in Campbell et al. (2002). Finally the "copolarized" and "crosspolarized" range corrected signal as delivered by the industrial software, respectively RCS_{co} and RCS_{cr} will be used as predictors for the machine learning algorithms described in part 3.

Raw data type and format depends on the instrumental device used. To make the algorithms usable on other devices, we used as input of the algorithms the files processed to a normalised format by the *raw2l1* routine. *raw2l1* is developed by the SIRTA and publicly available here: https://gitlab.in2p3.fr/ipsl/sirta/raw2l1

2.2 Radiosonde data

The algorithms are evaluated against radiosounding estimations. Meteo-France operates several radiosounding sites for the WMO Global Observing System. Two radiosoundings sites are collocated with the lidars of Brest and Trappes. They are equipped with a Meteomodem robotsonde and typically launch a Meteomodem M10 sonde at 11:15 AM and PM every day.

Many methods exists to derive BLH from radiosondes, and have been variously used in the literature.

- Parcel method: BLH is the height at which the θ profile reaches its ground value.
- Humidity gradient method: BLH is the height at which the gradient of relative humidity is strongly negative.
- Bulk Richardson number method: BLH is the height at which the bulk Richardson number exceeds 0.25 (threshold depending on authors).
- Surface-based inversion: BLH is the height at which the gradient temperature profile reaches zero.
- Stable layer inversion: BLH is the height at which the gradient of potential temperature profile reaches zero.

Hennemuth and Lammert (2006) use the parcel method and the humidity gradient methods. Collaud Coen et al. (2014) use all the techniques mentioned above and recommend the Bulk Richardson number method for all cases. Guo et al. (2016) use bulk Richardson number for a 2 years climatology. Seidel et al. (2010) compare the parcel method, the humidity gradient method, the surface-based inversion and other methods over 10 years and 505 sites over the world. Seidel et al. (2012) compare many methods and recommend bulk Richardson number.

After testing some of these methods on our dataset, we chose to derive boundary layer height with parcel method for the 11:15 sounding and bulk Richardson number for the 23:15 one.

2.3 Ancillary data

Ancillary data have been used for the description of the meteorological situation. They are not used by machine learning algorithms. All the instruments are collocated with lidar and radiosoundings.

- Rain gauges were used to detect rain events.
- Vaisala CL31 ceilometers were used to detect cloud base height, and distinguish cases with cloud on top or inside the boundary layer. Although the MiniMPL is perfectly capable of detecting clouds, we chose not to use the industrial algorithm. The algorithm has shown in our experience to make some false positives.
- Scatterometers were used to estimate visibility and detect fog cases.

3 Machine learning methods

Machine learning techniques are separated in two wide families: supervised learning (mimic a reliable reference) and unsupervised learning (learn without reference) (Hastie et al., 2009). First we present the supervised algorithm leading to ADABL. Second we present unsupervised learning leading to KABL.

3.1 Supervised learning method

Supervised methods learn from a reference. They are divided in two families: classification (aims to find the frontiers between groups) and regression (aims to approximate a function). In this work, we consider the boundary layer height derivation as a classification problem. From all points measured by the lidar, which are in the boundary layer and which are not? Then, the highest point of the boundary layer class is the BLH. Boosting algorithms are a very powerful family of algorithms, initially made for classification but they can also be used for regression (Hastie et al., 2009). AdaBoost (Adaptive Boosting) algorithm is well designed for binary classification (Freund and Schapire, 1997), thus it is the one we used.

3.1.1 AdaBoost algorithm

Let us consider the following problem: we have N vectors $x_i \in \mathbb{R}^p$ (here $p = 4$: seconds since midnight, height above ground copolarized channel, crosspolarized channel) and for each vector, we have a binary indicator $y_i \in \{-1, 1\}$ (-1 for boundary layer, 1 for free atmosphere). From the sample $(x_i, y_i)_{i \in [\![1,N]\!]}$, we want to predict the output indicator y_{new} of any new vector x_{new}. To do so, we must find a rule based on x_{new} coordinates values (the features) to cast it into the appropriate class. Decision

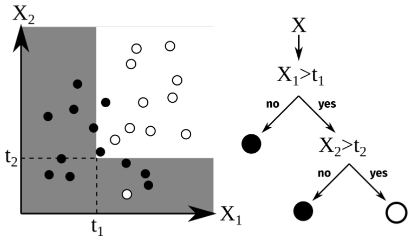

Figure 2. Illustration of binary classification with decision trees on fake 2-dimensional data

tree classifiers (Breiman et al., 1984) do this casting one feature at a time. For example, in the figure 2, there are black points and white points in a 2-dimensional space. Black points are mostly located where X_1 is low, hence the rule "if $X_1 < t_1$, then it is black". But in the other region, where $X_1 > t_1$, there are still some black points, all with low X_2. Therefore, at the output of the following rule, we add the rule "if $X_2 < t_2$, then it is black, else it is white." Decision trees are classifiers made up of such "if" statements, with various depths and thresholds. The deeper the tree, the more accurate the border, but the more complex the decision and the longer it is to train. As a matter of fact, deep trees are strongly subject to overfitting and they are less efficient than other methods. However, shallow decision trees are valuable because of their simplicity and their speed, even though their performance are quite limited (Hastie et al., 2009). They are often used as a *weak learner*: that is to say, classifiers with poor performances (although still better than random) but very simple (Freund and Schapire, 1997). In this study, weak learners in AdaBoost are trees with a maximum depth of 5 (maximum 5 forks between root an leaves).

AdaBoost is based on decision tree classifiers. It aggregates them in order to have the most accurate border. The idea of AdaBoost is illustrated by figure 3. First, a shallow decision tree is fitted on a random subsample of the dataset. Some points of the dataset are misclassified: the error of the classifier is the number of misclassified points. Another shallow decision tree is fitted on a subsample of the dataset where the previously misclassified points are over-represented. This new tree has new misclassified points, that will be over-represented in the training of the next tree, and so on, up the specified number of tree ($m = 200$ in our case). The classification given in output of AdaBoost is the average $\hat{y}$ of all the predicted class by the trees $\{\hat{y}_m\}_{m\in[\![1,M]\!]}$, weighted by their performance $\{\hat{\alpha}_m\}_{m\in[\![1,M]\!]}$: $\hat{y} = sign(\sum_{m=1}^{m} \hat{\alpha}_m \hat{y}_m)$

3.1.2 Training of the algorithm

Such algorithm must be trained from a trustworthy reference. For few days where the boundary layer is easily visible for a human expert, the boundary layer top is drawn by hand: all points below this limit are in the class "boundary layer", all points above are in the class "free atmosphere".

Two days were labelled by hand. These two days where chosen because the boundary layer is quite visible and they are in different site at different seasons. The first labelled day is a clear day of summer at Trappes, shown on figure 4 (top): a stable boundary layer is present near the ground at night, topped by a residual layer and few clouds between 02:00 and 04:00 UTC. The mixed layer starts developing at 9:00 UTC and stays around 2000 meters for the rest of the day. Around 22:00 UTC, a new stable layer seems to develop near the ground but it is not very clear where it starts and what is its extent. The second labelled

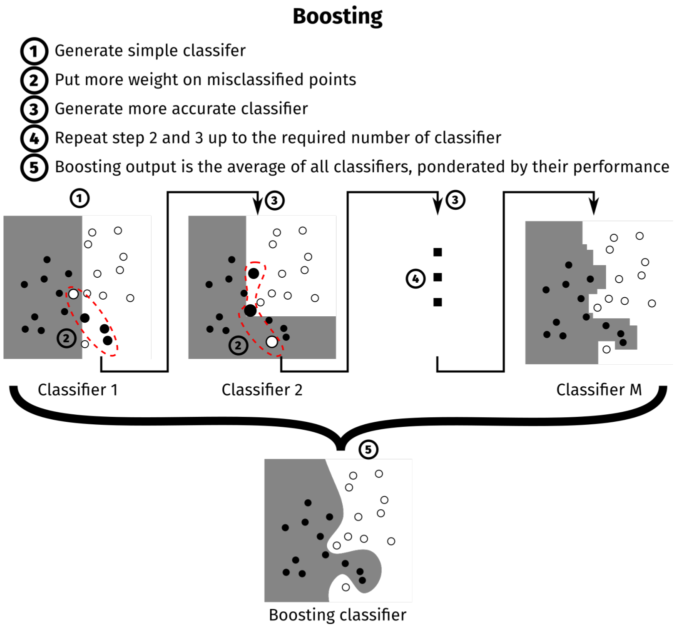
Boosting
1 Generate simple classifer
2 Put more weight on misclassified points
3 Generate more accurate classifier
4 Repeat step 2 and 3 up to the required number of classifier
5 Boosting output is the average of all classifiers, ponderated by their performance
1
2
3
3
4
Classifier 1
Classifier 2
Classifier M
5
Boosting classifier

Figure 3. Illustration of boosting on fake 2-dimensional data and 2 classes.

day is a clear day of winter at Brest, shown on figure 4 (bottom): a stable boundary layer is present near the ground at night, topped by a residual layer, shallower than at Trappes. The mixed layer starts developing at 9:00 UTC and stays around 1000 meters and decreasing along the day. Around 17:00 UTC, aerosols seems to accumulate in a thin layer close to the ground, therefore we choose to drop the BLH at that level.

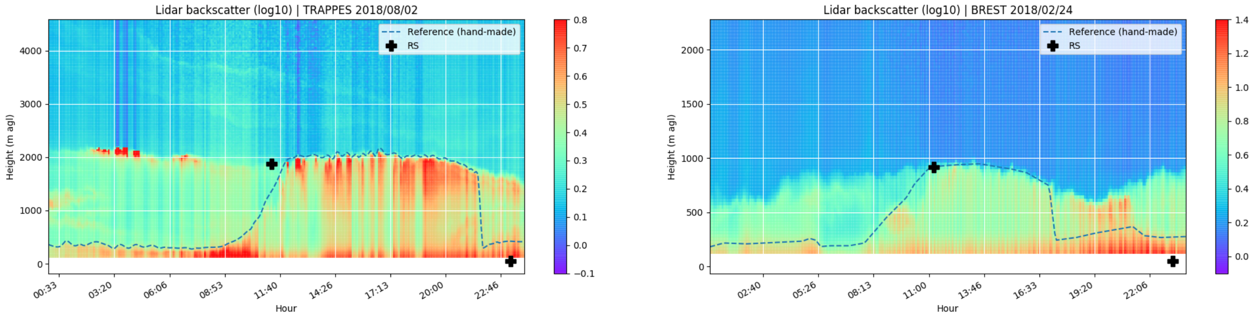

Figure 4. Hand-made reference and RS estimation over lidar range-corrected intensity signal for two days: 2nd of August 2018 at Trappes (top) and 24th of February at Brest (bottom).

The line of BLH made by hand is then loaded thanks to VGG Image Annotator software[1] to draw the BLH by hand and get the coordinates of the curve's points. Then, the output curve is interpolated with cubic spline to meet the lidar temporal resolution. Given the resolution of the lidar, this way of labelling the data gives $N = 86400$ individuals in total.

3.1.3 Retained configuration

Four predictors are used: the two lidar channels, the time (number of seconds since midnight) and the altitude (meters above ground level). The current configuration of ADABL is thus the following:

- Weak learner: decision tree of depth 5
- Number of weak learners: 200
- Predictors: time, altitude, RCS_{co}, RCS_{cr}

It was chosen by a trade-off between accuracy and computing time. The accuracy is the percentage of individuals well classified. It is estimated by cross-validation (random split, 80% training set, 20% testing set) and reaches 99.5% of the testing test. The computing is still reasonable: it takes 23 seconds to train on the full dataset and 3.7 seconds to predict the BLH for a full day.

It is possible to quantify the relative importance of the predictors (Breiman et al., 1984; Hastie et al., 2009). After the training, the time accounts for 30.3%, RCS_{co} for 28.4%, RCS_{cr} for 26.5% and the altitude for 14.8%.

3.2 Unsupervised learning methods

Unsupervised methods aim to find groups in data. In our case, we want to identify the group "boundary layer". The boundary layer height is then the border of this group. Two unsupervised learning algorithms have been tested: K-means and Expectation-Maximisation (EM).

3.2.1 K-means algorithm

The K-means algorithm is a well proven and commonly used algorithm to make data segmentation (Jain et al., 1999; Pollard et al., 1981). The algorithm has 3 steps:

1. Initialisation: K centroids $m_1, ..., m_K$ are initialized at random places inside the feature space.
2. Attribution: distances from all points to all centroids $(d(x_i, m_k))_{k \in [\![1,K]\!], i \in [\![1,N]\!]}$ are computed, and points are attributed to the closest centroid:
   $C(i) = arg\min_k \{d(x_i, m_k)\}$.
3. Update: centroids are re-defined as the average point of the cluster: $m_k = \frac{\sum_{i=1}^{N} x_i \mathbf{1}_{C(i)=k}}{\sum_{i=1}^{N} \mathbf{1}_{C(i)=k}}$

[1] Publicly available online following this URL: https://www.robots.ox.ac.uk/~vgg/software/via/via-1.0.6.html

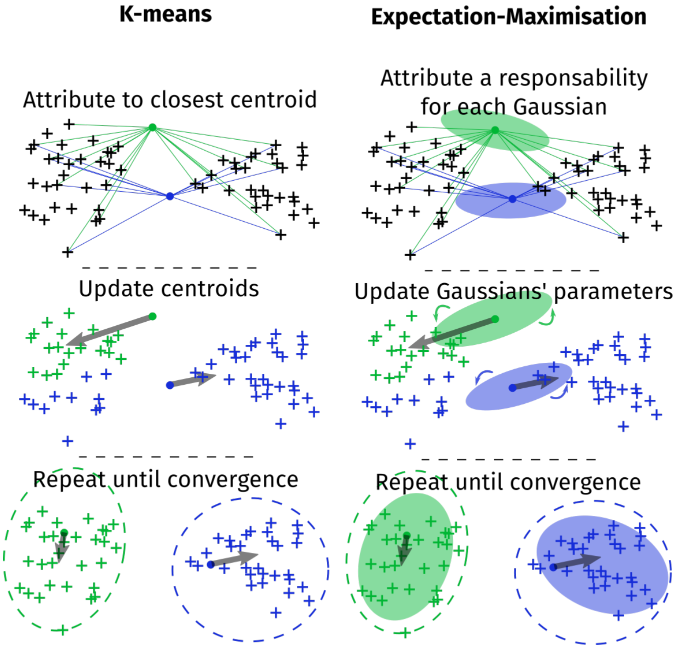
K-means
Expectation-Maximisation
Attribute to closest centroid
Attribute a responsability for each Gaussian
Update centroids
Update Gaussians' parameters
Repeat until convergence
Repeat until convergence

Figure 5. Illustration of K-means and EM algorithm on fake 2-dimensional data and 2 clusters.

Steps 2 and 3 are repeated until the centroids stop moving. It has been shown this algorithm converges to a local minimum of the intra-cluster variance Selim and Ismail (1984). The figure 5 (left) illustrates it.

190 ### 3.2.2 Expectation-Maximisation algorithm

The Expectation-Maximisation algorithm addresses the classification when the groups are Gaussian. It assumes each group $k \in [\![1, K]\!]$ is generated by a Gaussian distribution (μ_k, Σ_k). The algorithm estimates iteratively the parameters $\hat{\mu}_k$, $\hat{\Sigma}_k$ and the *responsibility* for each Gaussian $\hat{\gamma}_k^i$ (the *responsibility* is the probability for the point x^i to be generated by the k-th Gaussian). Points are then attributed to the group with the highest responsibility: $C(i) = arg\max_k(\hat{\gamma}_1^i, ..., \hat{\gamma}_K^i)$ The figure 5 (right) illustrates it. 195

K-means and EM algorithm are quite similar: if we assume all Gaussian have the same fixed variance and that this variance tends to zero, EM and K-means algorithms are the same. However, K-means does not rely on the Gaussian assumption of the groups.

3.3 KABL flowchart

200 The simplified flowchart of KABL is shown in figure 6. A netCDF file generated by the *raw2l1* software must be provided as input data of the KABL code. The data, namely: the vector of altitude z (size N_z), the vector of time t (size N_t), the range-corrected signals RCS_{co} and RCS_{cr} (matrices of shape $N_t \times N_z$), are extracted from this file. Such data are prepared

to fulfil machine learning algorithms requirements. For each time, **n_profiles** last profiles are extracted. Then the data they contain are normalised (remove mean and divide by standard deviation) and this provides a matrix X (shape $N \times p$) with $N =$**n_profiles**$\times N_z$ and $p = |$**predictors**$|$ (number of elements in the list). The matrix X is the usual input for machine learning algorithm: it has one line per individual observation and one column for each variable (or predictor) observed. For the need of BLH retrieval, the preparation provides also a vector Z (size N) with the altitude of each individual observation. The algorithm (either K-means or EM, the one specified in **algo**) is applied on the matrix X, with the parameters **n_clusters**, **init** and **n_inits**. It provides a vector of *labels* (size N) which contains the cluster attribution of each individual. Finally, we look for the first change in cluster attribution, starting from the ground. This gives us the BLH for this profile. These operations are repeated up to the end of the netCDF file.

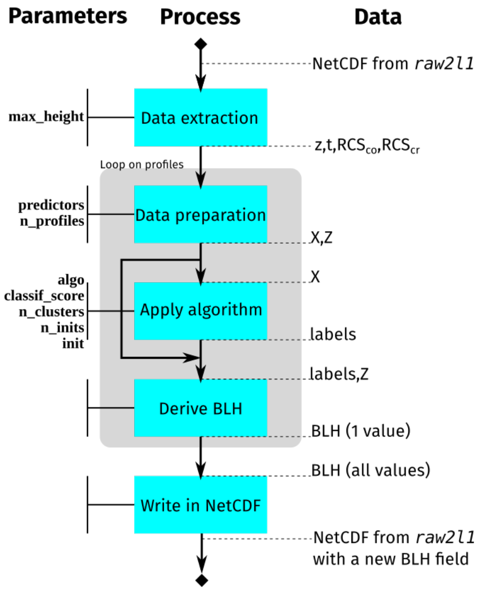

Figure 6. Simplified flowchart of KABL computer code

The parameters of this computer code were in bold font in the text and they are detailed here:

- **algo:** the machine learning algorithm that will be applied. Possible values are
  - 'gmm', for the EM algorithm (Gaussian mixture)
  - 'kmeans', for the K-means algorithm
- **classif_score:** internal score used to automatically choose the number of clusters (only used when n_clusters='auto'). See section 3.4 for a description of these scores.
- **init:** initialisation strategy for both algorithms. Three choices are available:
  - 'random': pick randomly an individual as starting point (both Kmeans and GMM)

- 'advanced': more sophisticated way to initialize (kmeans++ for Kmeans (Arthur and Vassilvitskii, 2007), the output a Kmeans pass for GMM)
- 'given': start at explicitly passed point coordinates

- **max_height:** height (meter above ground level) at which profiles are cut.

- **n_clusters:** the number of clusters to be formed (between 2 and 6). Either explicitly given, either determined automatically to optimise the score given in classif_score.

- **n_inits:** number of repetition of the algorithm. The more it is, the more likely is to find the global optimum, but also the more time it takes.

- **n_profiles:** number of profiles concatenated before the application of the algorithm. For example, if n_profiles=1, only the current profile is used. If n_profiles=3, the current profile and the two previous are concatenated and put in input of the algorithm.

- **predictors:** list of variables used in the classification. They can be different at night and at day. For both, it can be chosen among

  - RCS_{co}: copolarized range-corrected backscatter signal
  - RCS_{cr}: crosspolarized range-corrected backscatter signal

3.4 Performance metrics

Two kind of metrics have been used:

- External scores: they compare the result to a trustworthy reference. They have the advantage to give a meaningful evaluation of the performance, but they depend widely on the quality of the reference (accuracy and availability).

- Internal scores: they tell how well the classification is done, based only on the distances between points. They have the advantage to be always computable, but they are not linked to any physical property, hence are not always meaningful.

As none of them is perfect, the information brought by all give a broader understanding of the algorithms performance.

3.4.1 External scores

External scores use a reference to assess the quality of the result. In our case, the reference is the BLH estimated from RS and, when available, the BLH estimated by a human expert. If we denote by $\hat{Z}$ the estimated BLH (by any of the previously introduced algorithm) and by Z_{ref} the reference, the external scores used in this study are denoted as follow: the root-mean-squared error (E_2, equation 1), the average gap (E_1, equation 2), the Pearson's correlation (ρ, equation 3).

$$E_2 = \sqrt{\mathbb{E}\left[(\hat{Z} - Z_{ref})^2\right]} \tag{1}$$

https://doi.org/10.5194/amt-2020-78
Preprint. Discussion started: 7 April 2020

$$E_1 = \mathbb{E}\left[|\hat{Z} - Z_{ref}|\right] \tag{2}$$

$$\rho = \frac{cov(\hat{Z}, Z_{ref})}{\sigma(\hat{Z})\sigma(Z_{ref})} \tag{3}$$

In these formulae, $\hat{Z}$ and Z_{ref} are random variables. In the estimation of these scores, they are replaced by a vector of sample, and the expectation and standard deviation are replaced by their usual estimators. For unsupervised algorithms, such errors are calculated on all external information. For supervised algorithms, such errors are calculated only on the external information that was not used to train the algorithm (the test set) which is about 20% of the total.

3.4.2 Internal scores

The quality of a classification can be quantified by some scores only based on the labels and the distances between points. It gives an estimation of how trustworthy an estimation is, without external input. Many scores exist, with different formulation, different strengths and weaknesses (Desgraupes, 2013). In this study, three internal scores are used:

- Silhouette score (Rousseeuw, 1987).
  Compares average distance to its own group (a) to average distance to the neighbouring group (b): $S_{sil} = \frac{b-a}{\max(a,b)}$.
  1 is the best classification, 0 is neutral. -1 the worst.

- Calinski-Harabasz index (Caliński and Harabasz, 1974).
  Compares between-cluster dispersion (B) to within-cluster dispersion (W_k): $S_{ch} = \frac{(N-K)B}{(K-1)\sum_{k=1}^{K} W_k}$.
  $+\infty$ is the best classification, 0 the worst.

- Davies-Bouldin index (Davies and Bouldin, 1979).
  Compares the average distance to its group center ($\bar{\delta}_k$) to the distance between the group centers ($d(\mu_k, \mu_{k'})$): $S_{db} = \max_{k' \neq k} \left(\frac{\bar{\delta}_k + \bar{\delta}_{k'}}{d(\mu_k, \mu_{k'})} \right)$
  0 is the best classification, $+\infty$ the worst.

These three scores have been chosen to diversify the metrics and they are all implemented in Scikit-learn ($\geq$0.20).

3.4.3 Other metrics

In addition to the internal and external scores, the computation time and the number of invalid values (NaN or Inf) are recorded. Even though they don't measure how well the program is doing, they are useful for the user.

All the metrics used to measure the performance of KABL are summarized in the table 1.

https://doi.org/10.5194/amt-2020-78
Preprint. Discussion started: 7 April 2020

Metric	Type	Description	Best/worst value
corr	External	Pearson correlation coefficient	1/0
errl1	External	Average absolute gap with reference	0/+∞
errl2	External	Root-mean-squared gap with reference	0/+∞
s_score	Internal	Silhouette coefficient	1/-1
db_score	Internal	Davies-Bouldin index	0/+∞
ch_score	Internal	Calinski-Harabasz index	+∞/0
chrono	Other	Time to perform 24 hours of BLH estimation	0/+∞
n_invalid	Other	Number of invalid values (NaN or Inf) in 24 hours of BLH estimations	0/+∞

Table 1. Table of metrics used to measure the performance of KABL algorithm

4 Results

4.1 Sensitivity analysis of KABL algorithm

A sensitivity analysis was carried out on KABL code in order to find the "best" configuration. Various configuration of KABL have been tested extensively on a single day: the 2nd of August 2018 at Trappes, for which we have a hand-made reference (see figure 4-top). The more relevant configuration is then retained and tested on the two years dataset.

There are 8 parameters in the KABL code (see section 3.3 for their description). To assess the sensitivity of KABL to these parameters, the performance metrics (given in section 3.4) were estimated with the hand-made BLH as Z_{ref} and with KABL's output as $\hat{Z}$ for different combination of input parameters. The tested values for the input parameters are given in table 2 and the output metrics are given in table 1. We call a *configuration* a set of values for KABL parameters. Screening all the possible values listed in table 2 would take 3240 different configurations.

To look into these 3240 configurations at a glance, we started by estimating the influence of the code parameters (listed in table 2) onto the different metrics (listed in table 1). Their influence is quantified by first order Sobol indices (Sobol, 2001; Iooss and Lemaître, 2015; Rieutord, 2017), that is to say the ratio of the variance of the metric when the parameter is fixed over the total variance of the metric. If we denote by Y the metric and by X the vector of parameters, all considered as random, the first order Sobol index of the i-th parameter is defined by $S_i = V(\mathbb{E}[Y|X_i])/V(Y)$ (with $V(\cdot)$ denoting variance and $\mathbb{E}[\cdot]$ expectation). The higher the Sobol index, the higher the influence.

Figure 7 shows the Sobol indices obtained on the KABL computer code. Reading the matrix line by line, one can see that the metrics are sensitive to different parameters: silhouette score is very sensitive to **n_clusters**, for example, while Calinski-Harabasz index is sensitive to **n_profiles** and **predictors**. Reading the matrix column by column, one can see that some parameters are more influential than others (**classif_score** is much less than **n_clusters**, for example). It also highlights what

Parameter	Possible values	Meaning
algo	'kmeans'	The K-means algorithm is used
	'gmm'	The EM algorithm is used (Gaussian mixture model)
classif_score	'silh'	Silhouette score is used
	'db'	Davies-Bouldin index is used
	'ch'	Calinski-Harabasz index is used
init	'random'	Starting point are chosen randomly
	'advanced'	Starting points are chosen with smarter strategy
	'given'	Starting points are explicitly given
max_height	3500	Height (meter above ground level) at which profiles are cut
	4500	
n_clusters	2	Numbers of clusters to be formed is explicitly passed and is always the same
	3	
	4	
	5	
	'auto'	Automatically chosen to optimise **classif_score**
n_inits	10	Numbers of time the algorithm is repeated with different initialization (when **init** is not 'given')
	80	
n_profiles	1	Only the current profile is used
	2	The current profile and the previous are used
	3	The current profile and the 2 previous are used
	4	The current profile and the 3 previous are used
predictors	RCS0	Copolarized range-corrected signal is used all times
	dRCS0nRCS12	Copolarized range-corrected signal is used during daytime, both polarization channels are used separately during nighttime
	RCS12	Both polarization channel are used separately all times

Table 2. Table of possible values for the parameters of KABL computer code. With the dependencies between parameters, it gives 3240 different configurations.

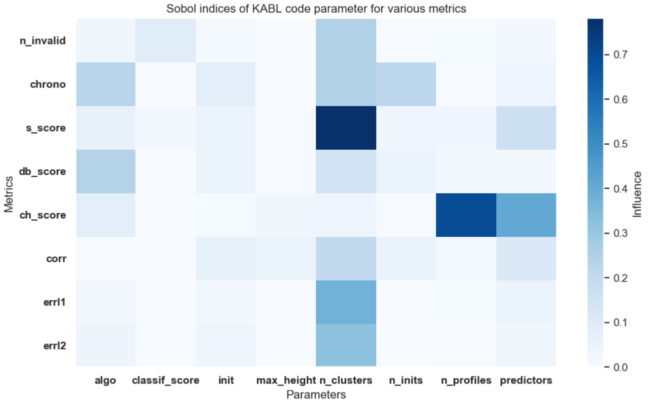

Figure 7. Relative influence on parameters over different metrics: on x-axis are the code's parameters, on y-axis are the metrics, the color shade represents the influence of the parameter over the metric.

295 are the main effects of changing this parameter, and hence, how to set it well. For each parameter, we will look at the metrics it has an influence on and decide which configuration is better.

Critical parameters are thus indicated in the figure 7 by the deepest blue columns, namely: **n_clusters**, **algo**, **predictors** and **init**[2]. For each of them, we have drawn in figure 8 the distribution (violin plots) of the relevant output conditionally to the parameter value. For example, the sub-figure 8-a has for abscissa the value of **algo** and for ordinate the computing time. The 300 3240 different configurations are divided in two parts: the ones with **algo**='kmeans' and the ones with **algo**='gmm'. What we see in sub-figure 8-a is the smoothed histogram of the computing time for the divided populations. The other sub-figures are constructed in the same way. Each line correspond to a critical parameter, and we represent the two most influenced outputs according to figure 7.

Parameters values are chosen to give the most optimal value for the metrics they have influence on. Optimal values are 305 indicated by a yellow star on each plot. To set **algo**, we better look at the computing time (8-a) and the Davies-Bouldin score (8-b): it results that 'kmeans' is the best choice for both (lower computing time and lower Davies-Bouldin index). To set **init**, we better look at the correlation (8-c) and the computing time (8-d): 'given' appear to be the best choice. To set **n_clusters**, we better look at the RMSE (8-e): 3 clusters is the best; and the silhouette score (8-f): 'auto' is the best. We chose to give the priority to the RMSE because silhouette score has also very high values for 2 clusters, which is suspicious. To set **predictors**, 310 we better look at the silhouette score (8-g) and the Calinski-Harabasz score (8-h): 'RCS0' appear to be the best choice. The same methodology was applied to the remaining parameters and the resulting configuration is in table 3. It will be used to generate the results in the next section.

[2] Although **n_profiles** has a large Sobol index for Calinski-Harabasz index, this influence is not explored because it is only due the increase of **ch_score** with the number of points.

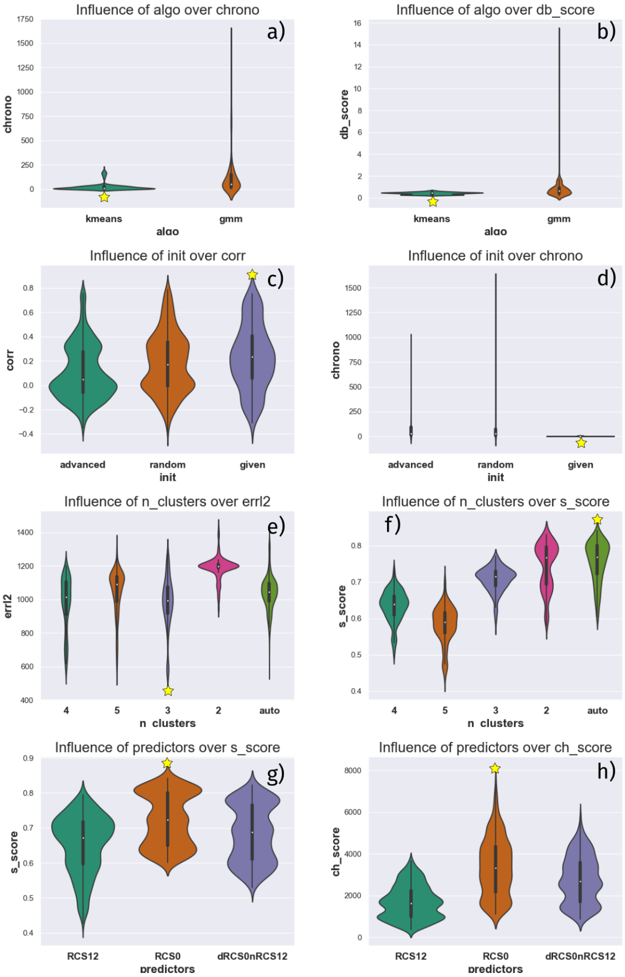

Figure 8. Distribution of the relevant outputs of the critical inputs: a) and b) show the effect of **algo** on computing time and Davies-Bouldin index (respectively). c) and d) show the effect of **init** on correlation and computing time. e) and f) show the effect of **n_clusters** on RMSE and silhouette score. g) and h) show the effect of predictors over silhouette score and Calinski-Harabasz score. For each sub-graph, the best parameter value is highlighted by an yellow star.

Parameter	Retained values
algo	'kmeans'
classif_score	'db'
init	'given'
max_height	4500
n_clusters	3
n_inits	10
n_profiles	1
predictors	RCS0

Table 3. Retained values for KABL computer code's parameters after sensitivity analysis

4.2 Two years comparison

All methods (KABL, ADABL and the manufacturer's) have been compared to radiosounding estimation for a two-year period.

4.2.1 Overall comparison

As explained in the section 3.4, three external scores are used to assess the quality of the estimations: the RMSE, the average gap and the correlation. As the average gap E_1 and the RMSE E_2 are very similar, we will show only the RMSE E_2 and the correlation ρ. In formulae 1 and 3, the reference BLH Z_{ref} is now the RS estimation, as described in section 2.2. To be able to compute such score, BLH estimation from lidar and from RS must be collocated. At the time of each RS estimation, the corresponding lidar estimation is the average of all available within the next 10 minutes after release (it means 1 or 2 lidar estimations). The following meteorological conditions have been discarded:

- Rain (rain gauge measures RR>0)
- Fog (scatterometer measures visibility <1000m)
- Low level cloud (ceilometer measures cloud base height <3000m)
- RS estimation below 120m (blind zone for lidar)
- Nighttime (launch of 23:15 UTC)

This selection rejects a large part of the dataset, but it ensures to keep only well-defined boundary layer. Meteorological conditions are measured by ancillary instruments presented in section 2.3. The results of the comparison are shown in figure 9.

In figure 9 we can see the results of the comparison between KABL and RS (blue bars), ADABL and RS (grey bars), manufacturer's algorithm and RS (orange bar). The first column represents the RMSE E_2 (the lower the better). The second column

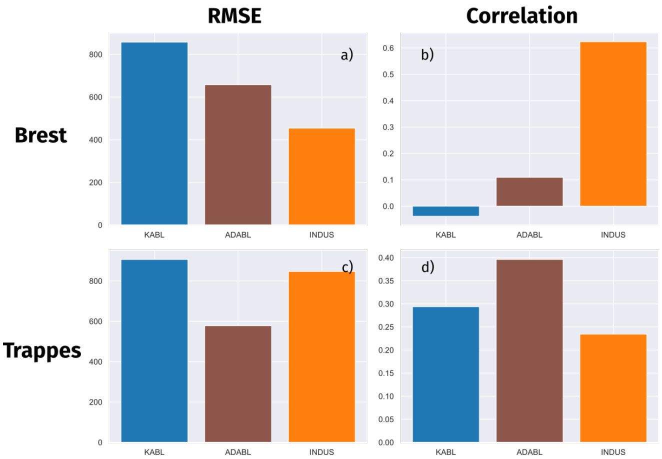

Figure 9. Results of the comparison with radiosounding for 2 years, on both sites, for two metrics: RMSE and correlation. Cases at night or with rain, fog, BLH estimated by RS under 120m or cloud under 3000m have been removed.

represents the correlation ρ (the higher the better). The first line shows the results for Brest site, the second line for Trappes site. KABL has the largest error for both sites, the lowest correlation for Brest and the second lowest for Trappes. ADABL has the best performance (highest correlation, lowest error) for Trappes and medium performances for Brest. Manufacturer's algorithm has the best performance for Brest, medium error and lowest correlation for Trappes. Overall, ADABL has better performances than KABL, and so has the manufacturer's. However, it is not clear which one between the manufacturer and ADABL is the best. Machine learning algorithms have better performances at Trappes. It may be because the boundary layer is better defined in Trappes than in Brest (generally higher, larger aerosol load and less often perturbed by synoptic disturbances).

4.2.2 Seasonal and diurnal cycles

In order to qualify the ability of the algorithm to give a consistent estimation of BLH, we have drawn in figure 10 the seasonal cycle (monthly average) and the diurnal cycle (six-minute average) on both sites. For each estimator, the thick line is the average BLH estimation and the shaded area in the inter-quartiles gap. Rain, fog and low clouds conditions have also been discarded. For monthly average, the night values are removed too. If we take only night estimation, the seasonal cycle is reversed for RS estimations: they are lower in summer. For other estimators, we do not see such a difference between day and night seasonal cycle (not shown).

In Brest (fig. 10-a), estimations from the manufacturer are lower than estimations from KABL and ADABL. Estimations from ADABL are usually higher than estimations from KABL (excepted in July). Radiosoundings give BLH estimate rather low in summer (June to October), rather high (higher than KABL's) in February and March and between manufacturer's and KABL's during the rest of the year. Overall, the manufacturer's has the cycle the closest to RS, while KABL overestimates and ADABL overestimates even more. Inter-quartiles distances (shaded areas) are large for all estimations, reflecting the wide range of values BLH estimations can take.

In Trappes (fig. 10-c) also, KABL and ADABL overestimate BLH in comparison to RS, while manufacturer's estimation is close. The seasonal cycle is more visible than in Brest: all BLH estimations are higher in summer than in winter. The one with the more marked cycle is KABL, while the one with the less marked cycle is RS. Inter-quartiles distances are also very large, especially in summer, because the difference between BLH at day and BLH at night is larger.

Figures 10-b and d show the diurnal cycle: all values with the same 0.1 hour in the day (6 minutes) were averaged. The diurnal cycle of RS cannot be drawn because they are only launched twice a day: at 11:15 and 23:15 UTC. However, we have drawn the average and quartile values at these hours, as a checkpoint for the other estimations. Manufacturer's and KABL estimations have both a quite smooth diurnal cycle, with lower BLH at night and a maximum around 15:00 UTC at Trappes and 13:00 at Brest. KABL's average is always higher than manufacturer's one. ADABL's estimation show a really different diurnal cycle, quite similar to the conceptual image we have of the boundary layer. Indeed, it has been trained on hand-made BLHs which reflects this conceptual image. Thus it is no surprise ADABL reproduce it well, however it may fail to adapt to special cases. It seems that the "time" predictor (number of second since midnight) has a large influence and is not balanced by other predictors. This is probably due to the fact it has been trained on only two days: sunrise and sunset this particular days and locations acquire an unbalanced importance. To balance this importance, the AdaBoost algorithm should be trained on more days and more sites, with a representative selection of cases.

4.3 Case study

The chosen case study is the 19th of April 2017 at Trappes. The boundary layer is clearly visible and has almost all the features of the conceptual image. It must be different than the days used for the training of ADABL.

In figure 11 is represented the co-polarized backscatter intensity (RCS_{co}) in shade of colors. Abscissa is the hour of the day (UTC). Ordinate is the height (meter above ground level). The different BLH estimates are superimposed in dotted lines: blue is KABL, orange is the manufacturer's algorithm, green is ADABL. At the beginning of the day, there is a thick residual layer with some plumes inside. Both KABL and manufacturer's include these plume into the boundary layer. Conversely, ADABL give a very low estimation where there is no visible frontier. In the morning (from 8 to 12 UTC), they all catch the transition reasonably well. However KABL has more irrelevant estimations (hitting what remains of the surface layer) than others, and ADABL goes too high with no apparent reason around 12:00. During the day, ADABL sticks to the boundary layer top, the manufacturer's sticks to the surface layer (quite visible) and KABL oscillates between both. The evening transition is blurry: the surface layer slowly sends back more and more signal to finally turn the mixed layer into a residual layer. KABL locates the transition very early (around 17:00), when it stops oscillating and sticks to the surface layer. ADABL makes the transition more smoothly, from 19:00 to 22:00. The manufacturer's algorithm is the latest to make the transition, around 23:00, quite sharply. We can conclude from this case study that no algorithm perfectly catches the boundary layer. Some of the limitations are physical: the evening transition is ill-defined, therefore algorithms disagree. The RS of 23:15 gives an estimation which is close to the lower boundary of lidar range. It highlights the fact that BLH below 120m are not that rare and cannot be detected with the lidar alone, whatever the method. Some others are algorithmic: KABL has an unfortunate trend to oscillate between

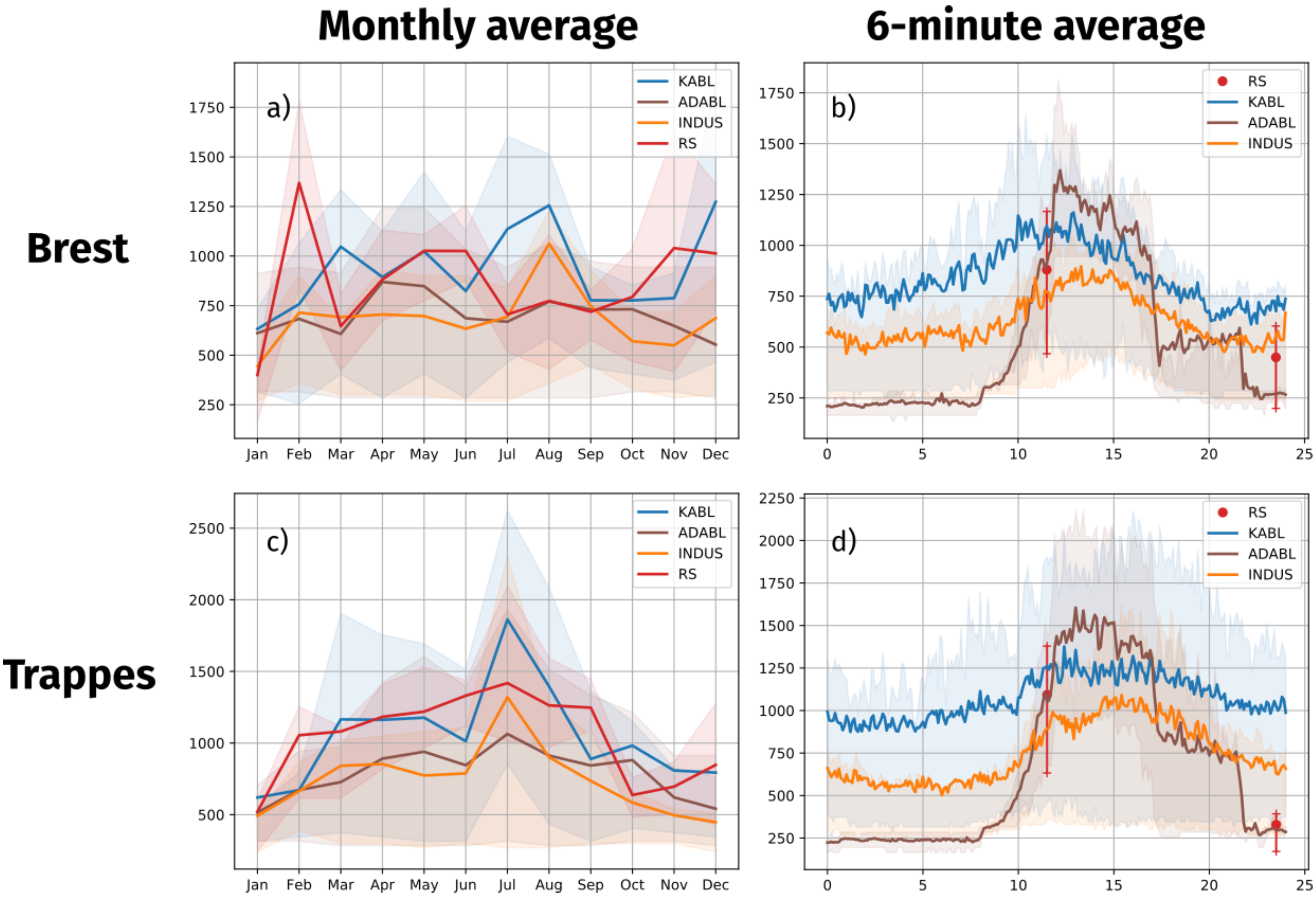

Figure 10. Seasonal (a and c) and diurnal (b and d) cycles of all BLH estimations on both sites. Thick lines represent the average, shaded area the quartiles.

several candidates for boundary layer top (surface layer or clouds), ADABL reproduces too much the features of the days it
385 has been trained on (night estimation and morning transition).

5 Discussion

This sections discusses various aspects of the results and the methodology. For the sake of readability, it has been split in many short paragraphs.

5.1 Algorithms maturity

390 Both algorithms are very recent. K-means algorithms have already been used to detect BLH in previous studies (Toledo et al., 2014, 2017; Rieutord et al., 2014). Therefore it is the most mature method and this is visible in this paper by the level of

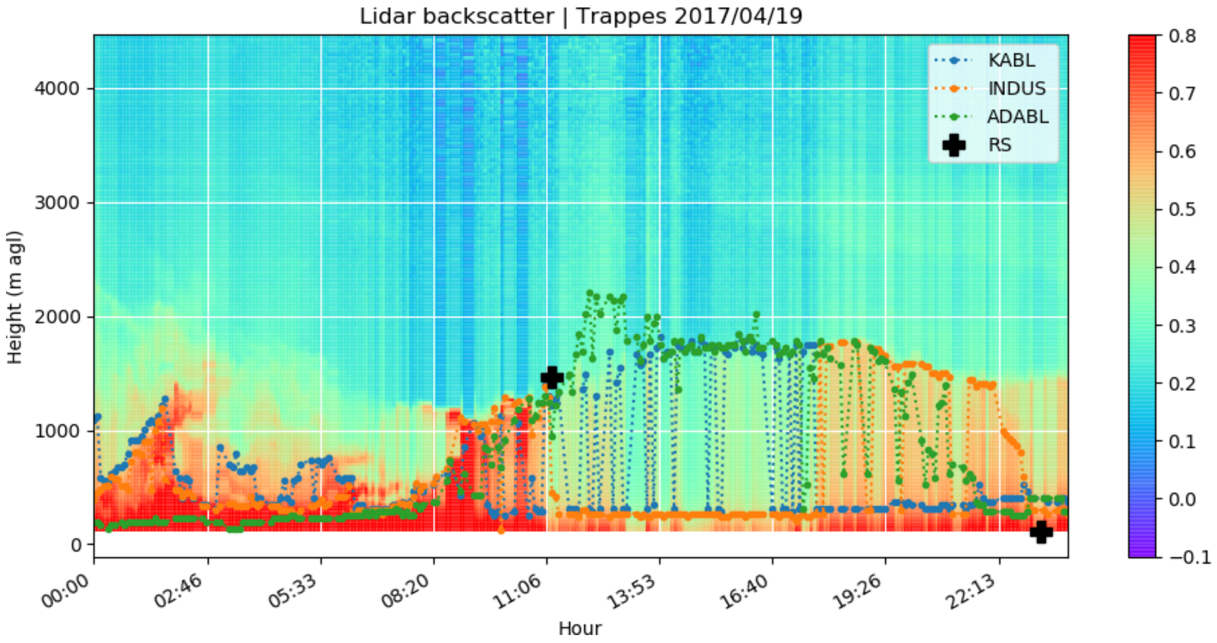

Figure 11. Case study: BLH estimations by different methods the 19th of April 2017 at Trappes.

investigation which was much higher for KABL than for ADABL. Concerning boosting, it is the first time, to our knowledge, that such algorithm is tested on this problem. Thus ADABL is a completely new algorithm and yet it outperforms KABL and competes with the manufacturer's. However, it raises training issues.

395 5.2 Real time estimation

Although it has not been necessary for this study, all algorithms can be used in real time: as soon as the backscatter profile is available, BLH estimations can be performed instantaneously[3]. However, it has been shown that KABL suffers from undesirable oscillations from one profile to the next. A method to filter these oscillations will be needed, but it can also divest the "real-time" property.

400 5.3 Quality of the evaluation

Although we had made an effort to sort meteorological conditions with ancillary data, the comparison over 2 years still mixes heterogeneous situations. More precise casting of meteorological situation (with atmospheric stability indices or large scale insight, for example) would lead to a better understanding of algorithm strengths and weaknesses.

[3] Both KABL and ADABL need less than 1 second to run on a single profile

5.4 Quality of the reference

Radiosoundings are unquestionably the reference for altitude measurements. However, the derivation of BLH from the measurements is more questionable, as several methods exists and (strongly) disagree. Moreover, they cannot be used to assess the full diurnal cycle of the BLH. It is a clear limitation of this study: it cannot tell if the difference between ADABL's diurnal cycle and the others is an improvement of not. Therefore a very interesting prospects would be to use dedicated field experiment with high frequency RS or other continuously running instruments as reference. For example, microwave radiometers are good candidates, as they bring an information not based on aerosols and BLH derivation is usual from these instruments (Cimini et al., 2013).

5.5 Training of ADABL

ADABL shows already good performances while it has been trained on two days only. It has been shown that most of its bad estimations come from the short length of its training period. Hence, a short term prospect would be to label more days, with various meteorological conditions. However, its dependence on training makes it more sensitive to instrumentation settings and calibration. Although the effect of a calibration or an evolution of the instrumental device has not been studied, it is likely that the training must be repeated after each calibration or change in the instrumental device. Therefore, two strategies are possible for the training of ADABL: remove the influence of calibration before training (it would require to know the instrumental constants for all devices) or train it to deal with differences (it would require to include as many different devices as possible in the training set, which would become very large). In any case, the main limitation will be the need to label all the dataset (a priori by human expert).

5.6 KABL is "trainingless"

KABL appeared to be the less well performing algorithm in this study, but it has very interesting ways of improvement. It does not require any training, therefore it is less dependent on instrumentation settings and calibration. As it is poorly dependent on instrumental device, one could use it on backscatter profiles made by other instruments (ceilometers). Moreover, one can imagine to add other profiles than backscatter intensity: after normalisation, it is just an additional predictor for unsupervised learning. Thus the idea of KABL can be pushed further to make instrument synergy between remote sensing instruments. Microwave radiometers are (again) good candidate, because they have a comparable time resolution and they bring an independent information on the thermal stratification of the boundary layer. Cloud radar have also comparable time resolution and they bring another independent information.

6 Conclusions

This paper has described two algorithms based on machine learning to estimate the mixing layer height from aerosols lidars measurements. One of them is based on K-means algorithm: it is named KABL (K-means for Atmospheric Boundary Layer).

The other is based on AdaBoost algorithm: it is named ADABL (AdaBoost for Atmospheric Boundary Layer). Both take the same input: one day of data generated by *raw2l1* routine; and give the same output: the time series of BLH for this day. KABL is a non-supervised algorithm. It will look for a natural separation in the basckscatter signals between the boundary layer and the free atmosphere. ADABL is a supervised algorithm. It fits a large number of decisions trees on a labelled dataset and aggregates them smartly to give a good prediction. KABL, ADABL and the lidar manufacturer's algorithm (unknown) have been tested on a 2-years dataset taken from Meteo-France operational network of lidars. The sites of Trappes and Brest have been chosen for their different climate and the availability of regular radiosoundings, which were used as the reference. KABL has the largest error for both sites, the lowest correlation for Brest and the second lowest for Trappes. ADABL has the best performance (highest correlation, lowest error) for Trappes and medium performances for Brest. Manufacturer's algorithm has the best performance for Brest, medium error and lowest correlation for Trappes. Overall, ADABL has better performances than KABL, and so has the manufacturer's algotithm. However, it is not clear which one between the manufacturer and ADABL is better. By analysing seasonal and diurnal cycles, we can see that KABL and manufacturer's estimations have similar behaviour, but KABL is always higher of about 200m. ADABL has the most marked diurnal cycle, with a look very similar to the expected diurnal cycle, but it depends too much on the days it has been trained on. Especially, the sunset and sunrise time of these days are over-influencing the estimations. On the case study, we can see that both algorithms perform globally well, but we have also illustrated some algorithmic limitations: KABL has an unfortunate trend to oscillate between several candidates for boundary layer top (surface layer or clouds), ADABL is too much constained by the days it has been trained on (night estimation and morning transition).

This experiment show that, despite few training and no maturity on the application of boosting on this problem, ADABL is competing with the manufacturer's algorithm. However, it is dependent on a trustworthy training set to get improved. Although KABL estimations do not always match with RS ones, it has the valuable advantage to not being dependent on a training set. Therefore it might easily be extended on backscatter profiles from other instruments (like ceilometers).

Code availability. The source code of KABL was made to be available and usable to all users, including commercial ones. It is freely available under an open-source license at the following link: https://github.com/ThomasRieutord/kabl It is made in Python 3.7 with regular packages in statistics and machine learning, namely: Scikit-learn 0.20 (Pedregosa et al., 2011) and SALib 1.3.7 (Herman and Usher, 2017), which are all open-source and under free licences. The repository contains all necessary features to run the code on *raw2l1* outputs. Several days of data are also provided as examples.

Author contributions. Tiago Machado implemented a first version of KABL's code and did the first comparisons with RS. Sylvain Aubert extracted and processed all the data (lidar, RS, ancillary), made one of the hand-labelled BLH and participated actively to the redaction of the manuscript. Thomas Rieutord implemented the current version of KABL and ADABL, made one of the hand-labelled BLH, produced the figures and participated actively to the redaction of the manuscript.

465 *Competing interests.* The authors declare that they have no conflict of interest.

Acknowledgements. The authors would like to thank Alexandre Paci, Alain Dabas and Olivier Traullé for their helpful reading and comments. Thanks to Marc-Antoine Drouin, from SIRTA for giving us the link to *raw2l1*. Thanks to all Meteo-France agents who install and sustain the lidar network.

References

Arciszewska, C. and McClatchey, J.: The importance of meteorological data for modelling air pollution using ADMS-Urban, Meteorological Applications, 8, 345–350, 2001.

Arthur, D. and Vassilvitskii, S.: k-means++: The advantages of careful seeding, in: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, pp. 1027–1035, Society for Industrial and Applied Mathematics, 2007.

Besse, P., Guillouet, B., and Laurent, B.: Wikistat 2.0: Educational Resources for Artificial Intelligence, arXiv preprint arXiv:1810.02688, 2018.

Breiman, L., Friedman, J., Olshen, R., and Stone, C.: Classification and Regression Trees, Wadsworth, 1984.

Brooks, I. M.: Finding boundary layer top: Application of a wavelet covariance transform to lidar backscatter profiles., Journal of Atmospheric & Oceanic Technology, 20, 2003.

Caliński, T. and Harabasz, J.: A dendrite method for cluster analysis, Communications in Statistics-theory and Methods, 3, 1–27, 1974.

Campbell, J. R., Hlavka, D. L., Welton, E. J., Flynn, C. J., Turner, D. D., Spinhirne, J. D., Scott, III, V. S., and Hwang, I. H.: Full-Time, Eye-Safe Cloud and Aerosol Lidar Observation at Atmospheric Radiation Measurement Program Sites: Instruments and Data Processing, Journal of Atmospheric and Oceanic Technology, 19, 2002.

Cimini, D., De Angelis, F., Dupont, J.-C., Pal, S., and Haeffelin, M.: Mixing layer height retrievals by multichannel microwave radiometer observations, 2013.

Cohn, S. A. and Angevine, W. M.: Boundary layer height and entrainment zone thickness measured by lidars and wind-profiling radars, Journal of Applied Meteorology, 39, 1233–1247, 2000.

Collaud Coen, M., Praz, C., Haefele, A., Ruffieux, D., Kaufmann, P., and Calpini, B.: Determination and climatology of the planetary boundary layer height above the Swiss plateau by in situ and remote sensing measurements as well as by the COSMO-2 model, Atmospheric Chemistry and Physics, 14, 13 205–13 221, 2014.

Davies, D. L. and Bouldin, D. W.: A cluster separation measure, IEEE transactions on pattern analysis and machine intelligence, pp. 224–227, 1979.

De Bruine, M., Apituley, A., Donovan, D., Klein Baltink, H., and de Haij, M.: Pathfinder: Applying graph theory for consistent tracking of daytime mixed layer height with backscatter lidar, Atmospheric Measurement Techniques Discussions, 10, 1893–1909, 2017.

Desgraupes, B.: Clustering indices, University of Paris Ouest-Lab Modal'X, 1, 34, 2013.

Dupont, J.-C., Haeffelin, M., Badosa, J., Elias, T., Favez, O., Petit, J., Meleux, F., Sciare, J., Crenn, V., and Bonne, J.: Role of the boundary layer dynamics effects on an extreme air pollution event in Paris, Atmospheric environment, 141, 571–579, 2016.

Flynn, C. J., Mendoza, A., Zheng, Y., and Mathur, S.: Novel polarization-sensitive micropulse lidar measurement technique, Opt. Express, 15, 2785–2790, 2007.

Freund, Y. and Schapire, R. E.: A decision-theoretic generalization of on-line learning and an application to boosting, Journal of computer and system sciences, 55, 119–139, 1997.

Gamage, N. and Hagelberg, C.: Detection and analysis of microfronts and associated coherent events using localized transforms, Journal of the atmospheric sciences, 50, 750–756, 1993.

Guo, J., Miao, Y., Zhang, Y., Liu, H., Li, Z., Zhang, W., He, J., Lou, M., Yan, Y., Bian, L., et al.: The climatology of planetary boundary layer height in China derived from radiosonde and reanalysis data, Atmospheric Chemistry and Physics, 16, 13 309, 2016.

Haefele, A., Hervo, M., Turp, M., Lampin J-L, Haeffelin, M., Lehmann, V., et al.: The E-PROFILE network for the operational measurement of wind and aerosol profiles over Europe, Proceedings of WMO Technical Conference on Meteorological and Environmental Instruments and Methods of Observation (CIMO TECO 2016, Madrid), 2016.

Haeffelin, M., Angelini, F., Morille, Y., Martucci, G., Frey, S., Gobbi, G., Lolli, S., O'dowd, C., Sauvage, L., Xueref-Rémy, I., et al.: Evaluation of mixing-height retrievals from automatic profiling lidars and ceilometers in view of future integrated networks in Europe, Boundary-Layer Meteorology, 143, 49–75, 2012.

Hastie, T., Tibshirani, R., and Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer Science & Business Media, 2009.

Hayden, K., Anlauf, K., Hoff, R., Strapp, J., Bottenheim, J., Wiebe, H., Froude, F., Martin, J., Steyn, D., and McKendry, I.: The vertical chemical and meteorological structure of the boundary layer in the Lower Fraser Valley during Pacific'93, Atmospheric Environment, 31, 2089–2105, 1997.

Hennemuth, B. and Lammert, A.: Determination of the atmospheric boundary layer height from radiosonde and lidar backscatter, Boundary-Layer Meteorology, 120, 181–200, 2006.

Herman, J. and Usher, W.: SALib: An open-source Python library for Sensitivity Analysis, The Journal of Open Source Software, 2, https://doi.org/10.21105/joss.00097, https://doi.org/10.21105/joss.00097, 2017.

Iooss, B. and Lemaître, P.: A review on global sensitivity analysis methods, in: Uncertainty management in simulation-optimization of complex systems, pp. 101–122, Springer, 2015.

Jain, A. K., Murty, M. N., and Flynn, P. J.: Data clustering: a review, ACM computing surveys (CSUR), 31, 264–323, 1999.

Kotthaus, S. and Grimmond, C. S. B.: Atmospheric boundary-layer characteristics from ceilometer measurements. Part 1: A new method to track mixed layer height and classify clouds, Quarterly Journal of the Royal Meteorological Society, 144, 1525–1538, 2018.

Krizhevsky, A., Sutskever, I., and Hinton, G. E.: Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, pp. 1097–1105, 2012.

LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning, nature, 521, 436–444, 2015.

Melfi, S., Spinhirne, J., Chou, S., and Palm, S.: Lidar observations of vertically organized convection in the planetary boundary layer over the ocean, Journal of climate and applied meteorology, 24, 806–821, 1985.

Menut, L., Flamant, C., Pelon, J., and Flamant, P. H.: Urban boundary-layer height determination from lidar measurements over the Paris area, Applied Optics, 38, 945–954, 1999.

Mohan, M., Bhati, S., Sreenivas, A., and Marrapu, P.: Performance evaluation of AERMOD and ADMS-urban for total suspended particulate matter concentrations in megacity Delhi, Aerosol and Air Quality Resarch, 11, 883–894, 2011.

Morille, Y., Haeffelin, M., Drobinski, P., and Pelon, J.: STRAT: An automated algorithm to retrieve the vertical structure of the atmosphere from single-channel lidar data, Journal of Atmospheric and Oceanic Technology, 24, 761–775, 2007.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E.: Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research, 12, 2825–2830, 2011.

Pollard, D. et al.: Strong consistency of k-means clustering, The Annals of Statistics, 9, 135–140, 1981.

Poltera, Y., Martucci, G., Collaud Coen, M., Hervo, M., Emmenegger, L., Henne, S., Brunner, D., and Haefele, A.: PathfinderTURB: an automatic boundary layer algorithm. Development, validation and application to study the impact on in-situ measurements at the Jungfraujoch, Atmospheric Chemistry and Physics Discussions, 2017.

Rieutord, T.: Sensitivity analysis of a filtering algorithm for wind lidar measurements, Ph.D. thesis, 2017.

Rieutord, T., Brewer, W. A., and Hardesty, R. M.: Automatic detection of boundary layer height using Doppler lidar measurements, 2014.

Rousseeuw, P. J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, Journal of computational and applied mathematics, 20, 53–65, 1987.

Seibert, P., Beyrich, F., Gryning, S.-E., Joffre, S., Rasmussen, A., and Tercier, P.: Review and intercomparison of operational methods for the determination of the mixing height, Atmospheric environment, 34, 1001–1027, 2000.

Seidel, D. J., Ao, C. O., and Li, K.: Estimating climatological planetary boundary layer heights from radiosonde observations: Comparison of methods and uncertainty analysis, Journal of Geophysical Research: Atmospheres, 115, 2010.

Seidel, D. J., Zhang, Y., Beljaars, A., Golaz, J.-C., Jacobson, A. R., and Medeiros, B.: Climatology of the planetary boundary layer over the continental United States and Europe, Journal of Geophysical Research: Atmospheres, 117, 2012.

Seity, Y., Brousseau, P., Malardel, S., Hello, G., Bénard, P., Bouttier, F., Lac, C., and Masson, V.: The AROME-France convective-scale operational model, Monthly Weather Review, 139, 976–991, 2011.

Selim, S. Z. and Ismail, M. A.: K-means-type algorithms: a generalized convergence theorem and characterization of local optimality, Pattern Analysis and Machine Intelligence, IEEE Transactions on, pp. 81–87, 1984.

Senff, C., Bösenberg, J., Peters, G., and Schaberl, T.: Remote sensing of turbulent ozone fluxes and the ozone budget in the convective boundary layer with DIAL and Radar-RASS: A case study, Contributions to atmospheric physics, 69, 161–176, 1996.

Sobol, I. M.: Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates, Mathematics and computers in simulation, 55, 271–280, 2001.

Stull, R. B.: An introduction to boundary layer meteorology, vol. 13, Springer, 1988.

Toledo, D., Córdoba-Jabonero, C., and Gil-Ojeda, M.: Cluster analysis: A new approach applied to lidar measurements for atmospheric boundary layer height estimation, Journal of Atmospheric and Oceanic Technology, 31, 422–436, 2014.

Toledo, D., Córdoba-Jabonero, C., Adame, J. A., De La Morena, B., and Gil-Ojeda, M.: Estimation of the atmospheric boundary layer height during different atmospheric conditions: a comparison on reliability of several methods applied to lidar measurements, International journal of remote sensing, 38, 3203–3218, 2017.

Ware, J., Kort, E. A., DeCola, P., and Duren, R.: Aerosol lidar observations of atmospheric mixing in Los Angeles: Climatology and implications for greenhouse gas observations, Journal of Geophysical Research: Atmospheres, 121, 9862–9878, 2016.