Base Price Algorithm: Documentation

Guenter Hitsch, Jihong Song

April 8, 2017

Contents

1 Introduction

This document explains the base price detection algorithm in detail. The algorithm has two parts–"base" and "promo"–that differ mostly in the initialization step. Section 2 discusses the initialization steps, section 3 explains the continuation steps, section 4 describes the adjustment steps, and 5 shows how those different steps are assembled into the two parts of our algorithm.

Caution: Most of the steps in our algorithm involve working with prices or base prices with in a *window*. It is possible that a window used in a step exceeds the time range of that step (typically

$[T_0 = 9, T_1 = T - 9]$, shown at the beginning of each step). Unless otherwise stated, in such cases we will just cut the window at the time limits.

2 Initialization

2.1 (Base) Set Base Price to Mode of Top Prices

The base algorithm initializes a price as a base price if it is equal to the model price in either a forward-looking or a backward-looking window.

Notation: Throughout the document, T_0 is set to 9 (`mode_T_before-1`) and T_1 is set to $T - 9$ (`T-mode_T_after+1`). The price series starts at time 0 and ends at time T.

Procedure For $t \in [T_0, T_1]$, find the modal prices in $[t - 9, t]$ and $[t, t + 9]$, breaking ties by maximum. If p_t equals one of the modal prices, set b_t to that model price.

Parameters `mode_T_before=10` and `mode_T_after=10` control window widths. They also define T_0 and T_1.

2.2 (Promo) Find Promoted and Not Promoted Prices

The promo algorithm categorizes each price in the series as non-promoted, promoted, or unlabelled. It then initializes base prices to be the non-promoted prices. This step proceeds in three parts. First we break the price series down to a sequence of "high" and "low" spells. Then we identify some high spells as "non-promoted prices" and some low spells as "promoted prices". Finally we remove false-positive non-promoted prices and relabel them as promoted prices.

Caution: The term "promoted price" in this section has a ***different*** definition from the same term in section 3.5.1.

2.2.1 Break the Price Series down to Spells

We define a "price spell" as a continuous series of prices that stay within 5% of the initial price. A spell is labelled "high" if it results from an upward break from the previous spell, and it is labelled "low" if it results from a downward break from the previous spell. The first spell starting from time 0 has no direction label.

Procedure We start from p_0 and call it the "reference price", p_{ref}. We then go forward in time and find the next t where $p_t < 0.95p_{\text{ref}}$ or $p_t > 1.05p_{\text{ref}}$, and we call the period t a starting point for a low or high spell, respectively. We then set $p_{\text{ref}} = p_t$ and repeat this process. The end point of a spell is defined to be the starting point of the following spell. In the end the whole price series will

be broken down to a collection of high and low spells, plus the initial spell starting p_0 that does not have a direction.

Parameters `threshold_factor_reference_decrease=0.95` says a price drop of more than 5% from the reference price triggers a new low spell.

`threshold_factor_reference_increase=0.95` says a price increase of more than 5% from the reference price triggers a new high spell.

2.2.2 Categorize Non-Promoted and Promoted Prices

We label a high spell as non-promoted prices if it is immediately followed by a low spell. Then we label the one or more low spells between two adjacent high spells as promoted prices if the price has dropped at least 10% from the median price of the first high spell.

Procedure *(Defining Non-Promoted Prices)* Let us call the starting points of directed spells $\{t_1 \cdots t_n\}$, and their directions $\{d_1 \cdots d_n\}$, where $t_1 > 0$ and $d_i = 1$ or -1 for high and low spells. We proceed if $n \geq 3$ (i.e. there are at least 3 significant price changes in the series) and find all k such that $d_k = 1$ and $d_{k+1} = -1$ (i.e. a pair of high and then low spells). Let K be the collection of such k and proceed if $K \neq \emptyset$. For each $k \in K$, we call $\{p_{t_k} \cdots p_{t_{k+1}-1}\}$ (i.e. all prices in the high spell) "*non-promoted prices*" and set them as base prices.

(Defining Promoted Prices) Let $K = \{k_1 < k_2 < \cdots < k_m\}$. For each $j = 1 \cdots m-1$, recall that $U_j = \{t_{k_j} \cdots t_{(k_j+1)} - 1\}$ is a high spell, $U_{j+1} = \{t_{k_{j+1}} \cdots t_{(k_{j+1}+1)} - 1\}$ is the next high spell, and the stuff in between, $D_{(j,j+1)} = \{t_{(k_j+1)} \cdots t_{(k_{j+1})} - 1\}$ consists of one or more low spells. We identify the prices in $D_{(j,j+1)}$ as "*promoted prices*" if the minimum price in it is at least 10% lower than the median price in U_j.

Parameters `threshold_factor_promotion=0.9` determines whether the low spell(s) between two high spells qualifies as a promotion.

2.2.3 Remove Mistaken Not Promoted Prices

A spell of non-promoted prices is consider "false-positive" if it is closer to the adjacent promoted prices than it is to the adjacent non-promoted prices. We re-label those false-positives as promoted prices.

Procedure For each $j = 1 \cdots m$, recall that $U_j = \{t_{k_j} \cdots t_{(k_j+1)} - 1\}$ is the j^{th} high spell. Let P_j^b be the set of at most 4 promoted prices in the window $[t_{k_j} - 13, t_{k_j} - 1]$ that are the closest to t_{k_j}. Let P_j^e be the set of at most 4 promoted prices in the window $[t_{(k_j+1)}, t_{(k_j+1)} + 12]$ that are the closest to $t_{(k_j+1)}$. Let N_j^b and N_j^e be the counterparts of P_j^b and P_j^e defined by non-promoted prices. Let $P_j = P_j^b + P_j^e$ and $N_j = N_j^b + N_j^e$[1].

[1] '+' here means list concatenation.

If P_j and N_j are both nonempty, let m_j^U be the median price in U_j, m_j^N be the median price in N_j, and m_j^P be the median price in P_j. Hence $m_j^N - m_j^P$ is a measure of the promotion magnitude around our spell of interest U_j. Now if $(m_j^U - m_j^P) \leq 0.5(m_j^N - m_j^P)$, we see that the prices in the spell U_j is closer to the set of nearest promoted prices P_j than it is to the set of nearest non-promoted prices N_j. In this case, we determine that the prices in U_j, previously categorized as non-promoted prices in Section 2.2.2, should actually be promoted prices. Therefore, we set the base prices in U_j to missing and remove the spell k_j from K.

Parameters `mistaken_not_promoted_T_window_before=13` and `mistaken_not_promoted_N_before=3` characterizes P_j^b and N_j^b.

`mistaken_not_promoted_T_window_after=13` and `mistaken_not_promoted_N_after=3` characterizes P_j^e and N_j^e.

`mistaken_not_promoted_ratio_threshold_promotion=0.5` characterizes what kind of non-promoted price spells should be removed.

3 Continuation

3.1 Continue Base Prices Within Tolerance

It a price is sufficiently close to an adjacent base price, identify it as a base price.

Procedure For $t \in [T_0, T_1]$, first, move forward in time. If $p_t \neq b_t$, b_{t-1} exists, and $|p_t - b_{t-1}| \leq 0.02$, then set $b_t = p_t$. Then, move backwards in time. If $p_t \neq b_t$, b_{t+1} exists, and $|p_t - b_{t+1}| \leq 0.02$, then set $b_t = p_t$.

Parameters `tol_dollars_lower=0.02` and `tol_dollars_higher=0.02` control the tolerance level for continuing base prices directly.

3.2 Find Valleys with Base Price

This step finds the "valleys" based on base prices. It starts from an existing base price (called "reference price") and looks forward for the next existing base price. The window between those two base prices qualifies as a "valley" if: (1) there are no more than 7 periods within the window, (2) the next base price is at least 0.9 times the reference price, (3) all prices in the window are smaller than one of the terminal base prices, and (4) at least one price in the window is smaller than 0.95 times the smaller terminal price.

For a valley, first set all base prices for periods within the window to the reference price. Then if the ending base price is higher than the reference price (i.e. there is a base price increase), we find the "adjustment period" by looking backwards from the end of window. Within the adjustment period, we set base prices to current prices.

Procedure *(Find a valley)* For $t \in [T_0 + 1, T_1]$, proceed if b_{t-1} exists and there is some $c \leq 7$ such that there is no base price in $[t, t + c)$, and b_{t+c} exists. (Replace $t + c$ by T_1 if $t + c > T_1$).

(Cover a qualified valley) If (1) $b_{t+c} \geq 0.9 b_{t-1}$, (2) $b_s \leq \max\{b_{t-1}, b_{t+c}\}$ for all $s \in [t, t + c)$, and (3) there exists some $s \in [t, t + c)$ such that $b_s \leq 0.95 \min\{b_{t-1}, b_{t+c}\}$, then set $b_s = b_{t-1}$ for all $s \in [t, t + c)$.

(Find adjustment period) If $b_{t+c} > b_{t-1}$, find $k \geq 0$ such that $p_{t+c-i} > b_{t-1}$ for all $0 \leq i \leq k$. Replace $b_{t+c-i} = p_{t+c-i}$ for each i from 0 to k.

Parameters `promotion_window=7` controls the maximum width of a valley (i.e. promotion). `next_base_price_percentage_factor=0.9` restricts that when the base price drops right after a promotion, the decrease cannot be higher than 10%. `min_base_price_percentage_factor=0.95` means that only a price drop of more than 5% from the smaller terminal price qualifies as a promotion.

3.3 Find Valleys with Price

This step finds the "valleys" based on observed prices. It starts from an existing base price (called "reference price") and looks forward for the next price that is larger than 0.95 times the reference price. This valley is then widened if (1) no base price is hit, (2) prices are below reference, and (3) prices are increasing. A valley qualifies as a promotion if (1) there are no more than 7 periods in the window, (2) each price in the window is smaller than one of the terminal prices, and (3) at least one price in the window is smaller than 0.95 times the smaller terminal price.

For a valley, first set all base prices for periods within the window to the reference price. Then if the ending base price is higher than the reference price (i.e. there is a base price increase), we find the "adjustment period" by looking backwards from the end of window. Within the adjustment period, we set base prices to current prices.

Procedure *(Find a valley)* For $t \in [T_0 + 1, T_1]$, proceed if b_{t-1} exists and there is some $c \leq 7$ such that c is the smallest integer satisfying $p_{t+c} \geq 0.95 b_{t-1}$. (Abort if $t + c > T_1$.)

(Widen the promotion valley) Find the largest integer k such that $c + k \leq 7$ and for all $s \in [t + c, t + c + k)$, (1) b_s does not exist, (2) $p_s \leq b_{t-1}$, and (3) $p_{s+1} \geq p_s$ are all satisfied. (Replace $t + c + k$ by T_1 if necessary.)

(Cover a qualified valley) If (1) $p_s \leq \max\{b_{t-1}, p_{t+c+k}\}$ for all $s \in [t, t + c + k)$ and (2) there exists some $s \in [t, t + c + k)$ such that $p_s \leq 0.95 \min\{b_{t-1}, p_{t+c+k}\}$, set $b_s = b_{t-1}$ for all $s \in [t, t + c + k)$.

(Find adjustment period) If $b_{t+c} > b_{t-1}$, find $k \geq 0$ such that $p_{t+c-i} > b_{t-1}$ for all $0 \leq i \leq k$. Replace $b_{t+c-i} = p_{t+c-i}$ for each i from 0 to k.

Parameters `promotion_window=7` controls the maximum width of a valley (i.e. promotion). `next_price_percentage_factor=0.95` restricts that the price at the end of the valley should not be more than 5% lower than the reference price. `min_price_percentage_factor=0.95` means that only a price drop of more than 5% from the smaller terminal price qualifies as a promotion.

3.4 Telescope

This steps looks forward and then backward searching for dots that can be connected. For example, in the forward-looking telescope we start from a base price (the "reference") and search forward in time (for no more than 12 periods) for an observed price that equals to the reference price. If the resulting window contains no base price, we just fill it with the reference price. The rationale here is that assuming the reference base price is correct, if the store charges the same price at a point t in a neighborhood of the reference period t_R, then the base prices between t and t_R should all be the reference base price. The implicit assumption here is that base prices tend to be stable within the specified window width (for example, in our application the forward window should not be longer than 12 weeks).

Procedure *(Forward)* For $t \in [T_0+1, T_1]$, proceed if b_{t-1} exists. Find the largest $k \in [1, 12]$ such that $p_{t+k} = b_{t-1}$ and $\{b_t \cdots b_{t+k-1}\}$ are all missing. Set $\{b_t \cdots b_{t+k-1}\}$ all to b_{t-1}.

(Backward) For $t \in [T_0, T_1 - 1]$, proceed if b_{t+1} exists. Find the largest $k \in [1, 3]$ such that $p_{t-k} = b_{t+1}$ and $\{b_{t-k+1} \cdots b_t\}$ are all missing. Set $\{b_{t-k+1} \cdots b_t\}$ all to b_{t+1}.

Parameters `telescope_window_forward=12` is the maximum window width for the forward telescope. `telescope_window_backward=3` is the maximum window width for the backward telescope.

3.5 Fill Gaps If Contains Promoted Prices

A base price gap is defined as a window in which base prices are all missing except at the two ends. We want to fill the gap with the reference price – the base price at the front end – if the gap contains promoted prices.

First we need to determine whether the prices within the gap are promoted. Since we don't know the base prices in the gap, we can only determine this based on our knowledge of promoted price levels before and after the gap. The `isProm` does that job for us.

3.5.1 The `isProm` function

Procedure Let us call the start and end periods t_s and t_e. Find the minimum price $p_{\min}$ and the mode price p_{mode} within the window $[t_s, t_e]$ (choose the minimum if there are ties in the modes).

Find up to 4 promoted prices (a price p_t is considered "promoted" if b_t exists and $p_t < b_t$) that are the closest to t_e in the window $[t_e + 1, t_e + 26]$, and let R be the set of those prices. Let r_1 be the first promoted price in the window, and let $r_{\min}$ be the minimum price in R. Return true if all the following conditions are satisfied:

(1) R is not empty

(2) r_1 differs no more than 5% from p_{mode}

(3) r_1 differs no more than 5% from p_{t_e}

(4) $p_{\min} < 1.2 r_{\min}$

Repeat the test on t_s and the window $[t_s - 26, t_s - 1]$ if the above test fails.

Parameters `comparison_promotion_window=26` is the maximum window width we use to find promoted prices before and after the "gap" $[t_s, t_e]$. `N_most_recent=4` is the maximum length of R that affects the calculation of $r_{\min}$. `promotion_percent_similarity_closest=0.05` and `promotion_comparison_factor=1.2` makes sure that the promoted price levels before or after the gap is not too far away from the observed prices within the gap (without this condition the interpolation cannot be justified).

3.5.2 The actual filling function

Procedure For $t \in [T_0 + 1, T_1]$, proceed if b_{t-1} exists and b_t does not exist. Find the first $1 \leq k \leq 26$ such that b_{t+k} exists. If no such k exists and $t + 26 > T_1$, set k such that $t + k + 1 = T_1$.

If `isProm` determines that there are promoted price in the window $[t, t+k-1]$, then set $\{b_t \cdots b_{t+k-1}\}$ all to b_{t-1}.

Parameters `gap_window=26` sets the maximum allowed length of a gap to be filled.

4 Adjustment

4.1 Remove Loner Base Prices

This step removes a base price if it is charged no more than 4 times within a 21-week window.

Procedure For $t \in [T_0, T_1]$, proceed if b_t exists. Count the number of occurrence N of b_t within $[p_{t-10} \cdots p_{t+10}]$. Set b_t to missing if $N \leq 4$.

Parameters `loner_T_before=10` and `longer_T_after=10` control the window width for counting the number of occurrence. `loner_N=4` controls what qualifies as a loner base price that should be removed.

4.2 Find Adjustment Periods

This step finds intervals between base prices where prices are either monotonically increasing or monotonically decreasing for at most a certain number periods, and records the prices in these periods as base prices.

Procedure For $t \in [T_0 + 1, T_1 - 1]$, proceed if b_{t-1} exists and b_t is missing. Find the smallest c such that $\{b_{t+1} \cdots b_{t+c}\}$ are all missing. If $\{b_{t-1}, p_t, \cdots p_{t+c}, b_{t+c+1}\}$ is an increasing sequence and $c \leq 1$, or if the sequence is decreasing and $c \leq 1$, then set $\{b_t \cdots b_{t+c}\}$ to $\{p_t \cdots p_{t+c}\}$.

Parameters `N_adjustment_periods_up=1` and `N_adjustment_periods_down=1`.

4.3 Remove Isolated Base Price Clusters

This step detects continuous clusters of base prices of at most length 1. If there are no base prices before the cluster within a window of at least length 1 and and if there are no base prices after the cluster within a window of at least length 1, then we infer that the cluster is "isolated" and does not represent actual base prices.

Procedure Initialize $\{c_t\} = 0$ as a boolean series that tracks whether we have changed the base price at each period. For each $t \in [T_0 + 1, T_1]$, proceed if b_t is missing and b_{t-1} exists or $c_{t-1} = 1$.

Let $B \geq 0$ be the smallest number so that b_{t+B+1} exists and $\{b_{t+1} \cdots b_{t+B}\}$ are all missing.

Let $S \geq 1$ be the smallest number so that $b_{t+B+S+1}$ is missing and $\{b_{t+B+1} \cdots b_{t+B+S}\}$ all exist.

Let $A \geq 1$ be the smallest number so that $b_{t+B+S+A+1}$ exists and $\{b_{t+B+S+1} \cdots b_{t+B+S+A}\}$ are all missing.

If $B+1 \geq 1$, $S \leq 1$, and $A \geq 1$, remove all base prices in $\{b_{t+B+1} \cdots b_{t+B+S}\}$, and set $\{c_{t+B+1} \cdots c_{t+B+S}\}$ all to 1.

Parameters `min_T_isolated_before=1` controls the minimum number of missing base prices before the cluster. `min_T_isolated_after=1` controls the minimum number of missing base prices afte the cluster. `cluster_T_max=1` controls the maximum size of the isolated cluster.

4.4 Correct Mistaken Base Prices

This step finds valleys that qualify as promotions, and corrects any base price within the valley to the reference base price before the valley.

We start from a base price drop of more than 5% within 4 weeks (say from period t_1 to t_2). We then check whether the base price recovers at least 75% of this drop in an additional 10 weeks (if so, call the period of recovery t_3). If the `isProm` function determines that the window $[t_1, t_3]$ qualifies as a promotion, then any previously inferred base prices in this window should be wrong, and they are all corrected to the reference price – i.e. the base price at t_1.

Procedure For each $t \in [T_0, T_1]$, proceed if b_t exists. Let c be the smallest number where b_{t+c} exists and $\{b_{t+1} \cdots b_{t+c-1}\}$ are all missing. Proceed if $c \leq 4$, and repeat the process for $t \leftarrow t + 4$ otherwise.

Proceed if $b_{t+c} < 0.95 b_t$. Let $d \geq 2$ be the smallest number where $b_{t+c+d} \geq 0.75 b_t + 0.25 b_{t+c}$. Proceed if $d \leq 10$.

If the window $[t, t+c+d-1]$ contains promoted prices (determined by the `isProm` function discussed in Section 3.5.1), then any base price within this window is determined mistaken and corrected to b_t. Repeat the process for $t \leftarrow t + c + d - 1$.

Parameters `mistaken_window_for_initial_ps_drop=4` and `mistaken_initial_threshold_factor=0.95` limit the initial base price drop to at least 5% within 4 weeks.

`mistaken_window=10` and `mistaken_end_threshold_factor=0.75` require that the base price recovers at least 75% of the initial drop within 10 weeks.

5 Combining All These Parts

The whole algorithm runs in a hybrid way. We run the base algorithm and promo algorithm separately. We only use the output of the promo algorithm in periods where the base algorithm does not identify a base price.

5.1 Base Algorithm: Steps

1. Set base price to mode of top prices
2. Remove loner base prices
3. Continue base prices within tolerance
4. Find valleys with base price
5. Find valleys with price
6. Continue base prices within tolerance
7. Find adjustment periods
8. Remove isolated base price clusters
9. Forward telescope
10. Backward telescope
11. Continue base prices within tolerance
12. Correct mistaken base prices
13. Fill gaps if contains promoted prices
14. Correct mistaken base prices

5.2 Promo Algorithm: Steps

1. Find promoted and not promoted prices
    - Abort if no non-promoted price is found
2. Continue base prices within tolerance
3. Find valleys with base price
4. Find valleys with price

5. Continue base prices within tolerance
6. Find adjustment periods
7. Forward telescope
8. Backward telescope
9. Continue base prices within tolerance
10. Correct mistaken base prices
11. Fill gaps if contains promoted prices
12. Correct mistaken base prices