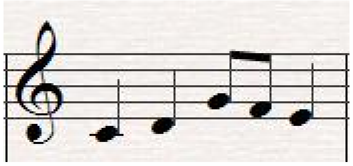
How the original algorithm works - a 5 note example

Original piece:



Let's say our reference trajectory is:

C – 1
D – 2
G – 0
F – 2
E – 3

The program will sort these values by the following method:

1. Take the first number and compare it to the second number. If the first number is greater, switch them.
2. Take the first number (which may have been the second number before) and compare it to the third number. If the first number is greater, switch the two.
3. Continue comparing all the numbers to the first number in this fashion.
4. Take the second number and compare it to the third number. If the second number is greater, switch the two.
5. Continue comparing all the numbers above the second number to the second number, and repeat the process for the third number, the fourth number, and so forth.

[1(C), 2 (D), 0(G), 2(F), 3 (E)]

The first number is greater than the third number, so they are switched.

[0(G), 2 (D), 1(C), 2(F), 3 (E)]

There are no more numbers greater than the first number.
The second number is greater than the third number, so they are switched.

[0(G), 1(C), 2(D), 2(F), 3 (E)]

The numbers are now in order.

Now let's say the variation trajectory is [0.5, -1, 2, 3, 4]

For each number in the variation trajectory, the program checks it against the numbers in the reference trajectory, starting at the right side where the greatest numbers are. Once it finds a number in the reference trajectory that is less than the variation trajectory number, it takes the number to the right of it and assigns the note associated with that reference value to the variation.

For example, the first number in the variation trajectory, 0.5:
0.5 ≤ 3 (E) – Keep looking
0.5 ≤ 2 (F) – Keep looking
0.5 ≤ 2 (D) – Keep looking
0.5 ≤ 1 (C) – Keep looking
0.5 > 0 (G) – Assign the number to the right, 1, and its associated note, C.

For the second number, -1:
-1 ≤ 3 (E) – Keep looking
-1 ≤ 2 (F) – Keep looking
-1 ≤ 2 (D) – Keep looking
-1 ≤ 1 (C) – Keep looking
-1 ≤ 0 (G) – Keep looking
Reached the end – assign the lowest number, 0, and its associated note, G.

For the third number, 2:
2 ≤ 3 (E) – Keep looking
2 ≤ 2 (F) – Keep looking
2 ≤ 2 (D) – Keep looking
2 > 1 (C) – Assign the number to the right, 2, and its associated note, D.

For the fourth number, 3:
3 ≤ 3 (E) – Keep looking
3 > 2 (F) – Assign the number to the right, 3, and its associated note, E.

For the fifth number, 4
4 > 3 (E) – Since there is nothing to the right, assign the original fifth note, E.

So the result would be C, G, D, E, E.

In the previous example, there were two notes with the same reference trajectory value. In this case, the D, which came first in the song, also came first in the list of trajectory values after they had been sorted. With this sorting algorithm, this is not always the case. Let’s look at another example using the same song with a different reference trajectory.

C – 1
D – 2
G – 3
F – 2
E – 0

Now, let’s apply the sorting algorithm.

[1(C), 2(D), 3(G), 2(F), 0(E)]

The first number is greater than the fifth number, so they are switched.

[0(E), 2(D), 3(G), 2(F), 1(C)]

There are no more numbers greater than the first number.

The second number is greater than the fifth number, so the two are switched.

[0(E), 1(C), 3(G), 2(F), 2(D)]

There are no more numbers greater than the second number.

The third number is greater than the fourth number, so the two are switched.

[0(E), 1(C), 2(F), 3(G), 2(D)]

There are no more numbers greater than the third number.

The fourth number is greater than the fifth number, so the two are switched.

[0(E), 1(C), 2(F), 2(D), 3(G)]

The numbers are now in order. However, the result is different from the previous example in that E and G are switched (because we switched their trajectory values) and that the 2 representing F has switched to the left side of the 2 representing D. If we take the same variation trajectory as the previous example, we can see what kind of effect this has on the variation that is created.

Variation trajectory = [0.5, -1, 2, 3, 4]

For the first number in the variation trajectory, 0.5:
0.5 ≤ 3 (G) – Keep looking
0.5 ≤ 2 (D) – Keep looking
0.5 ≤ 2 (F) – Keep looking
0.5 ≤ 1 (C) – Keep looking
0.5 > 0 (E) – Assign the number to the right, 1, and its associated note, C.

For the second number, -1:
-1 ≤ 3 (G) – Keep looking
-1 ≤ 2 (D) – Keep looking
-1 ≤ 2 (F) – Keep looking
-1 ≤ 1 (C) – Keep looking
-1 ≤ 0 (E) – Keep looking
Reached the end – assign the lowest number, 0, and its associated note, E.

For the third number, 2:
2 ≤ 3 (G) – Keep looking
2 ≤ 2 (D) – Keep looking
2 ≤ 2 (F) – Keep looking
2 > 1 (C) – Assign the number to the right, 2, and its associated note, F.

For the fourth number, 3:
3 ≤ 3 (G) – Keep looking
3 > 2 (F) – Assign the number to the right, 3, and its associated note, G.

For the fifth number, 4
4 > 3 (G) – Since there is nothing to the right, assign the original fifth note, E.

So the result would be C, E, F, G, E.
This is quite different from the previous variation, C, G, D, E, E, partly because the values for E and G were switched, and partly because the location of the D and the F in the sorted trajectory were switched, causing F to be picked instead of D. Such switches of identical trajectory values can have interesting results, but it is very difficult to predict when such switches will occur. If you want to prevent such switches, use a bubble sort instead of the sort described here.