隨機播放不隨機

鍾廷楷
聖公會白約翰會督中學

最近剛剛教完高中機會率，有一天某同學告訴我在網上看到這篇文章，他想理解當中的意思。先看看文章內容：

圖 1

「如果你在聽歌時按下了 #隨機播放
卻接連聽到同歌手同專輯甚至同順序的歌曲
你可能就會懷疑：這隨機功能是否有問題？
當你有這個想法時，就是犯了 #賭徒謬論

機率的思考方式，大多會違逆我們的直覺
賭徒謬論就在指出，每個事件發生的機率
並不會受到前一次事件的影響
但我們總以為，只要過去已發生過很多次
那接下來發生的可能性就會因此降低

這就像是當你丟硬幣連丟出 99 次正面後
第 100 次丟出反面的機率依舊會是 50%
但一定會有人拿全部身家賭下次出反面

同樣地，前面才剛播完離開地球表面
緊接著就私奔到月球的機率並不會受影響

只是我們總會誤以為，所謂的隨機播放
就應該是不同專輯不同歌手不同曲風
以平均分布的方式一一出現

就因為這種對「隨機」的錯誤想像
（據說真的有很多人會跟客服抱怨）
導致現在的線上串流，大多會修正演算法
減少隨機播放時類似歌曲重複出現的機率

在多了這層人為干涉後
所謂的隨機，其實就不是真正的隨機了…」[4]

1. 賭徒謬論(The Gambler's Fallacy)

對於同學來說，賭徒謬論可能比較陌生。賭徒謬論是一種涉及機會率的謬誤，指人類對於曾經發生過很多次的事件，會偏向低估再下一次發生的機會，又或是連續很久沒有發生的事件，偏向估計它越來越大機會發生。例如引文中擲硬幣的例子，又或是六合彩賭博中，人們偏向覺得久未出現的數字，出現機會越來越大。

但在播放歌曲軟件的隨機播放中，如果「接連聽到同專輯同歌手」又是否屬於賭徒謬論的例子呢？筆者初想時也是如此認為，但細心思考下，發覺作者有點誤解賭徒謬論的最重要假設。原因在於，賭徒謬論不適合於非獨立事件（Non-independent events）。隨機播放一般是指將音樂庫內的歌曲按隨機次序播放，在播放完所有歌曲前並不會重複同一首歌曲。假如現在音樂庫內有 30 首歌，其中 5 首是甲歌手的，其他 25 首是其他歌手。按下隨機播放後，第一首歌就是甲歌手的歌曲，那麼第二首亦是甲歌手的歌的機會是多少？答案當然是 4/29。但如果第一首不是甲歌手，那麼第二首是甲歌手的歌的機會是多少？答案是 5/29。由於相同歌曲在同一次播放內不會重複，第一首歌是哪一首實際上會影響第二首歌出現哪一首的機會，即這兩件事並非是獨立事件。所以如果第一首是甲歌手，第二首是相同歌手的歌曲的確是受影響地小了，是故賭徒謬誤在這個例子並不成立。

4 以上文字及圖片，取自 incrediville_tw 的 Instagram。
https://www.instagram.com/p/Bi4SWwEn6gs/?taken-by=incrediville_tw

如果一定要用賭徒謬論去解釋隨機播放，可以將例子改為「多次隨機播放時，仍然發生甲歌手的歌曲連續播放」這個例子。如果你按隨機播放，出現了甲歌手歌曲連續播放。如果幸運地／不幸地，你 50 次隨機播放後，每次均出現了甲歌手歌曲連續播放，在第 51 次隨機播放時，出現相同情況的機會會否少了?答案是此事件的發生機會率與你第一次播放時發生的機會率一樣。如果你覺得之前 50 次已經出現了甲歌手歌曲連續播放，第 51 次仍是這樣的機會率將會很少，那麼你就是犯下了賭徒謬論。

2. 隨機播放不隨機？

筆者在看完上述網上文章後很感興趣，於是在網上搜尋了一些關於隨機播放的討論文章。其中一篇[5] 內容相當有趣，是指某大網上音樂平台的隨機播放很不隨機，很多用家投訴即使某些歌手或風格的歌在曲庫內不算多，這些歌手或風格的歌曲仍然經常連續地出現。

筆者腦海中立即浮現出兩個想法，第一，隨機播放的曲序是如何生成？這個問題會留待第 3 部份去了解，第二，如果歌曲播放的次序是真正隨機，出現兩首相同歌手或曲風連續播放，亦即是上部份末段事件的機會率應該如何計算？

為暫時方便討論，用回上述的例子，曲庫內有 30 首歌曲，5 首屬於甲歌手，其他 25 首來自不同歌手。在本部份中，請假設所有隨機播放的曲序都是真正隨機。真正隨機意指所有可能的曲序是均等機會出現，30 首歌曲的無限制排序為 30!，約有 2.65×10^{32} 不同的排法。

要計算甲歌手歌曲連續出現的機會率前，不妨先計算它的互補事件的機會率（Complementary Events）。在某次歌曲清單內，沒有甲歌手歌曲連續播放的機會是多少？請讀者先猜猜這個機會，是多於 50%還是少於 50%？一點題外話，這類排除連續出現情況的排列數目計算，在香港新高中數學課程推行時，引入了核心課程部份。

---

5 Griffin, A. (2015, February 24). 'Random' shuffle on your music player is tricking you. Retrieved from https://www.independent.co.uk/life-style/gadgets-and-tech/news/why-random-shuffle-feels-far-from-random-10066621.html

其中一個想法如下，先考慮 25 首其他歌曲排序 $= 25!$，而 5 首甲歌手由於要分隔出現，共有 26 個位置可供選擇。同學可想像成是 25 首歌曲之間的空隙，包括最頭及最尾位置。26 個位置放 5 首歌曲兼具次序的排列 $= \mathrm{P}_5^{26}$，故合符條件的總數為 $25! \times \mathrm{P}_5^{26}$。

如果沒有任何限制，30 首歌共有 30! 的排列數目。

所以機會率 $\mathrm{P}(\text{沒有甲歌手歌曲連續播放}) = \frac{25! \times \mathrm{P}_5^{26}}{30!} \approx 0.4616$。

觀看算式，會發現 $\frac{30!}{25!} = \mathrm{P}_5^{30}$，上述算式可簡化為 $\frac{\mathrm{P}_5^{26}}{\mathrm{P}_5^{30}}$，可理解為其他 25 首歌的出現次序無關痛癢，只考慮那 5 首歌的位置，任意放置時有 30 個位置可供考慮，有限制時就只能放在那 26 個分隔空隙上。

甲歌手的歌曲佔整個曲庫才約 16.67%，但在完全隨機的情況下，只有大約 46.16%的情況才會出現沒有連續甲歌手歌曲播放的曲序！換句話說，在某次隨機播放中，有約超過一半（53.84%）機會會出現甲歌手歌曲連續播放的情況。仔細一點分析，這裡指的甲歌手歌曲連續播放有很多不同的情況，例如出現 1 次 2 首甲歌手連續，或是出現 1 次 4 首甲歌手連續播放，又或是同時出現 1 次 2 首和 1 次 3 首的連續播放情況。有興趣的讀者，可以嘗試列出所有符合 5 首歌連續播放的情況，對大家了解排列組合有一定的幫助。

現在嘗試計算其中一種情況：在某次隨機播放中，出現僅有 1 次的 2 首甲歌手連續播放的機會是多少？

用回剛才的思路，先安排 25 首其他歌曲排序，即 25!；之後在 5 首甲歌手歌曲中，選取 2 首作連續播放，但由於 2 首歌次序不同都要計算一次，所以有 P_2^5 的取法；在 26 個歌曲空隙中，揀選 1 個去放那 2 首連續歌曲（C_1^{26}）；最後在剩下的 25 個空隙中選 3 個，放剩下的 3 首甲歌手歌曲，並有次序（P_3^{25}）。

$$\mathrm{P}(\text{出現 1 次的 2 首甲歌手連續播放}) = \frac{25! \times \mathrm{P}_2^5 \times \mathrm{C}_1^{26} \times \mathrm{P}_3^{25}}{30!} \approx 0.4196$$

原來仍有四成多的機會，果真是「一切都不算得罕有」[6]。

一般的情況下，設曲庫內有 m 首歌曲，其中有 n 首為某同一歌手的歌曲，而且 $n \leq \left\lfloor \frac{m+1}{2} \right\rfloor$，沒有這位歌手的歌曲連續播放的機會將為 $\frac{P_n^{m-n+1}}{P_n^m}$ 。

下圖顯示了 30 首歌曲中，甲歌手的歌佔了幾多首與其相關不會連續播放的機會率表示。如圖 2 所示，來自某一歌手的歌曲佔 5 首或以上，不會連續播放的機會率已經低於 0.5。圖 3 則為總曲目為 100 首時的情況。

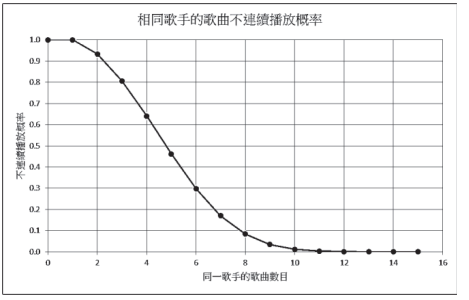

圖 2

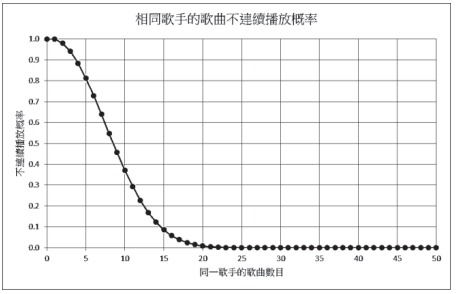

圖 3

總括而言，假如你的曲庫內，某個歌手／風格／大碟的歌曲如有一定數量的話，在真正隨機的情況，出現連續播放的情況是常見情況。舉一例子，現有 100 首歌曲在曲庫內，有 10 首甲歌手的歌曲，那麼你每次隨機播放時，聽到甲歌手歌曲連續播放的機會就高達 62.87%！而在現實情況中，

6 林夕（1998）。《我甚麼都沒有》。香港：華星唱片出版有限公司。

每位歌手可能都會有數首歌在曲庫中，出現相同歌手連續播放的機會就會更多。

回到整個討論核心，對於一般人來說，隨機播放的曲序如何才算是真正隨機？其中一個指標就是：「隨機播放的曲序中，相同歌手的歌曲應該很小機會連續播放」。但在剛才的討論中可見，只要相同歌手的歌曲佔總曲目的一定百分比，在真正隨機的情況下，完全沒有相同歌手連續播放的曲序才是比較少機會率出現的情況。

所以這個故事的結局，就是該音樂播放網站的工程師放棄了真正隨機的播放曲序，改為使用將相同特性（歌手、曲風、大碟等）的歌曲約略等距分隔開的播放算法[7]，使用家覺得這才是「真隨機」。真正隨機的，原來在大眾眼中並不隨機，刻意分佈的反而才是隨機，很精彩吧？

3. 隨機播放演算法

這一部份較為接近計算機科學的範疇，但經歷前述的討論，可能讀者仍有興趣了解更多。筆者會在此部份簡介基本的隨機播放演算法。

先旨聲明，本部份並不打算討論如何生成一個隨機數字，或是生成這個隨機數字的方法是真正隨機（true random）還是偽隨機（pseudorandom）。在此我們會假設我們能夠隨機地產生一個由 1 至 N 之間的隨機正整數，當然 N 亦為一正整數。

坊間大部份的隨機播放演算法，都是建基於由 Ronald Fisher 及 Frank Yates 在 1938 年發明的 Fisher-Yates 演算法，此演算法能夠產生一個由 1 至 N 的隨機排列次序。步驟如下：

1. 寫下由 1 至 N 的整數。
2. 記下未處理的數字數目 M，在 1 至 M 內取一個隨機整數 K。
3. 由數列開頭數起，取出第 K 個數字並放在新的表內。
4. 繼續步驟 2，直到所有數字已被處理。
5. 在步驟 3 得出的數字表就是所需要的隨機排序。

7 Poláček, L. (2014, December 09). How to shuffle songs? Retrieved from https://labs.spotify.com/2014/02/28/how-to-shuffle-songs/

下列是 7 個數字的例子示範：

步驟	1 – *M*	隨機數字	未處理數字	所生成隨機數列
1	7	5	1 2 3 4 ~~5~~ 6 7	5
2	6	4	1 2 3 ~~4~~ ~~5~~ 6 7	5 4
3	5	4	1 2 3 ~~4~~ ~~5~~ ~~6~~ 7	5 4 6
4	4	2	1 ~~2~~ 3 ~~4~~ ~~5~~ ~~6~~ 7	5 4 6 2
5	3	1	~~1~~ ~~2~~ 3 ~~4~~ ~~5~~ ~~6~~ 7	5 4 6 2 1
6	2	1	~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~ 7	5 4 6 2 1 3
7	1	1	~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~ ~~7~~	5 4 6 2 1 3 7

在 1964 年 Richard Durstenfeld 開發了電腦版的隨機排序法，雖然他沒有提及(或他可能不知道)，他的方法實在是建基於 Fisher-Yates 演算法的，而此改良版演算法更適合在電腦程式語言上實現。此改良版最主要不是剔出數字，而是將此隨機選出的數字與尾數作換位，並將尾數不斷向前遞增。原來的 Fisher-Yates 演算法中的第三步，如果原封不動的指令電腦執行，就需要不斷計算尚有多少剩餘的數字未被剔出及其相應的位置，而新的換位算法相比起來使用的記憶體及所需時間會少很多。用回上述的例子：

步驟	1 – *M*	隨機數字	所生成隨機數列 （尾數位置以 | 表示）
1	7	5	1 2 3 4 **7** 6 | **5**
2	6	4	1 2 3 **6** 7 | **4** 5
3	5	4	1 2 3 **7** | **6** 4 5
4	4	2	1 **7** 3 | **2** 6 4 5
5	3	1	**3** 7 | **1** 2 6 4 5
6	2	1	**7** | **3** 1 2 6 4 5
7	1	1	**7** 3 1 2 6 4 5

註： 如抽出的數字剛好是尾數，數列排序將會不變。例如 1 2 3 6 7 | 4 5，如果下一次抽出隨機數字 5，則 7 的位置不變，下一次的數列為 1 2 3 6 | 7 4 5。

此兩種方法能夠生成任意排序的隨機數列，而只要隨機數字沒有偏頗，得出不同數列的機會亦是均等的。在大部份現代的電腦語言當中，要執行上述演算法多數只需三數行的程式碼，相當容易使用。

但隨著前部份的討論以及大眾對「隨機」的看法，現在很多的播放程式未必會直接使用上述簡單直接的隨機演算法，而會先加上不同的形式的演算，讓隨機數列看來更「隨機」。但在前述的某音樂平台例子，在將曲目按分類平均分隔開後，平台仍會在每個小區間內使用這個簡潔的 Fisher-Yates 隨機演算法改良版去生成隨機曲目呢！

參考文獻

Durstenfeld, R. (1964). Algorithm 235: random permutation. *Communications of the ACM*, *7*(7), 420.

Fisher, R. A., & Yates, F. (1949). Statistical tables for biological, agricultural and medical research. *Statistical tables for biological, agricultural and medical research* (3rd edn.). Edinburgh: Oliver and Boyd.

作者電郵：tkchung@skhbbss.edu.hk