## 3.3对应点和对应轨迹段

本节主要介绍对应点以及基于对应点产生的对应轨迹段。一个样本点的对应点指的是该样本点到另一条轨迹中距离最短的点。有些算法中限制一个点的对应点必须为样本点，比如DTW，而一些算法则没有这个限制，如BDS（Bi-directional mapping similarity）算法。下面分别介绍这两种算法在寻找对应点时的优势与不足。

### 3.3.1BDS算发找对应点的优势与不足

之前的一篇论文中提到了双向映射相似性算法BDS。大致思想是，在二维空间中，有一条查询轨迹Q和一条数据轨迹R，首先找到轨迹Q的所有样本点到轨迹R上的对应点，以及轨迹R的所有样本点在轨迹Q上的对应点。这里一个点的对应点指的是该点对齐到另一条轨迹上最接近的位置，这个位置就是该点的对应点，并且这个位置不一定是另一条轨迹的某一个样本点，可能只是两个相邻样本点之间连线上的点。

有时候由于采样策略的原因，导致样本点的分布很不均匀，如图一所示。如果使用轨迹相似性计算中的欧氏距离算法，即采用样本点一一对应的关系的话，会导致样本点对应的时候错位很严重，不仅前后不同时间的样本点对应错误，更会由此导致整体距离变大，不能准确的描述原本空间、时间和形状都很接近的轨迹的相似程度。

如果采用BDS算法寻找对应点，则会直接将样本点匹配到另一条轨迹中距离自己最近的点，如图二所示。

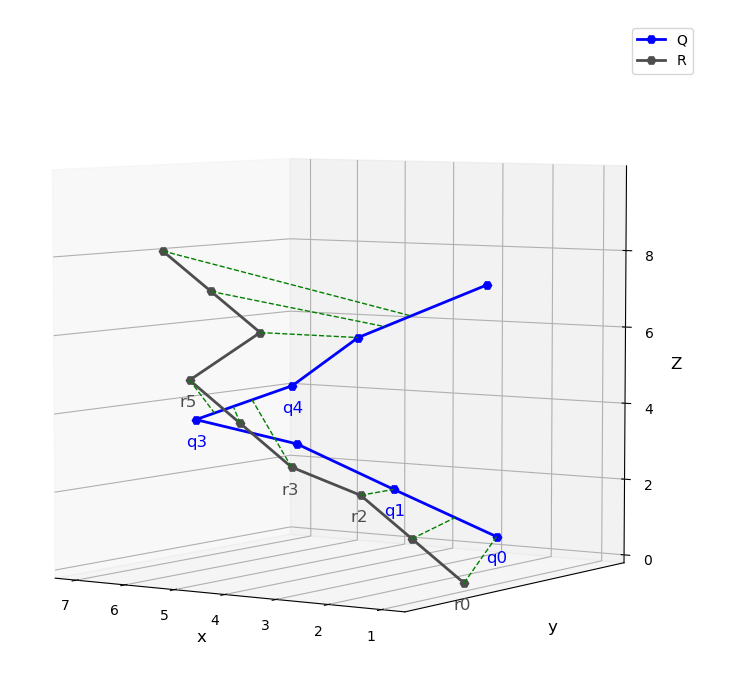
BDS的优点在于不考虑一个点的对应点必须要是样本点，减少了由于样本点位置和序列带来的限制，会更好地描述样本点之间的对应情况。





BDS算法使用了一种很好的样本点对应方法，但是BDS算法没有考虑时间因素以及轨迹方向因素，而本文研究的问题是在一个三维时空下，需要考虑时间的先后顺序以及轨迹的方向。如果将BDS算法直接使用在本场景中，有可能出现这样的情况，即轨迹R上的某个点在对应到轨迹Q上的一个点之后，时间戳在后面的点有可能对应到时间戳在前面的点。

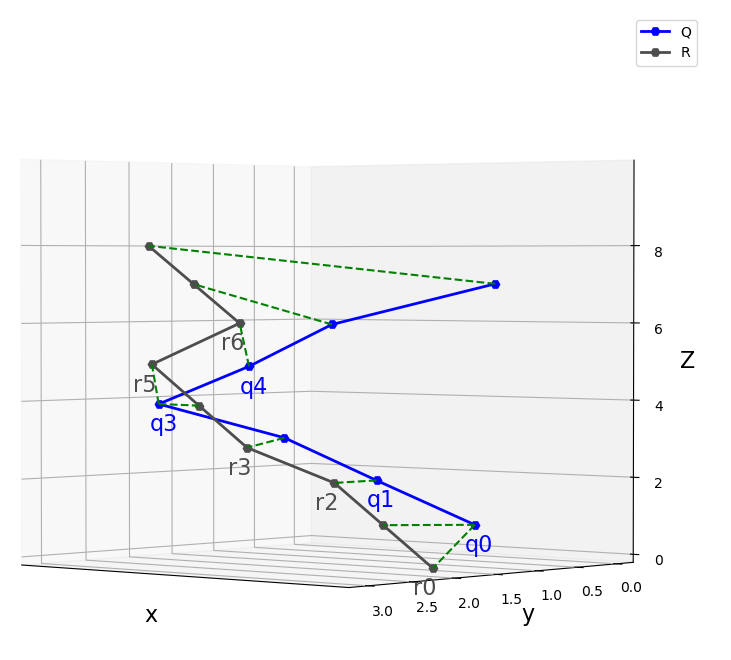
举个例子说明一下。如图一所示，先展示二维空间下的轨迹Q和R，其中R的至部分轨迹段和Q很相似，至部分轨迹段与Q形状相似，但方向相反。图二展示了加上时间维度的三维空间下的Q和R，由于空间表示不方便，所以隐藏了部分点的表示，图中可以看出对应点的关系，如果使用BDS中的方法，一个样本点到另一条轨迹距离最短的位置为该样本点的对应点的话，会导致轨迹R的至都与轨迹Q很相似，但是中间发生了行驶方向的翻转，对应样本点时序的错位，即至对应至，至对应至，因此与轨迹R进行比较的轨迹序列为<>，而不是我们输入的查询轨迹Q，虽然样本点都相同，但是一条时间序列还需要关注样本点时间上的先后顺序。由于BDS中寻找对应样本点的方法中没有考虑到同一条轨迹上样本点的时间先后，不能直接用于本文讨论的问题。

### 3.3.2DTW算法找对应点的优势与不足

在DTW算法中使用的样本点对齐方法很容易可以解决刚才BDS中产生的时序错位问题。针对样本点对齐问题，DTW是这么做的。首先计算轨迹Q和轨迹S的第一个节点之间的距离，然后加上下面三个距离的最小值，第一个是Q的第二个节点开始到S的DTW距离，第二个是Q到S的第二个节点开始的DTW距离，第三个是Q的第二个节点开始到S的第二个节点的开始的DTW距离。这种做法严格按照轨迹数据中样本点的时间顺序来往下计算，最后获得的轨迹间距离的值是所有对应点匹配方案中的最优方案贡献出来的，并且可以获得最优的对应点匹配方案。DTW算法肯定比欧氏距离算法中的使用样本点一一对应的方法要好很多。

如果使用DTW解决刚才的问题，那么其对应关系如图所示。样本点对应关系为：和对应，对应，对应，和对应，完全按照时间先后顺序来对应，在至处没有出现Q上对应点时序的翻转，不仅解决了BDS中时序错位的问题，还得到了一个让对应点之间总体距离最小的匹配方案。



因此DTW算法的优点是在对应点匹配过程中不会出现时序的错乱。但是由于DTW限制了只能是样本点之间的对应，这其中必然存在有些对应关系不是最优的，如果考虑使用BDS算法中的思想，对应策略还有优化的空间。

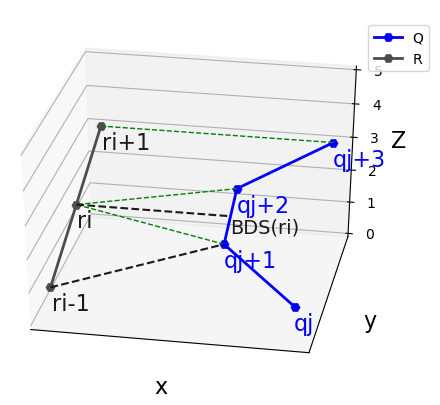
### 3.3.3DTW-BDS对应点匹配算法

基于上述对DTW和BDS算法的讨论，发现二者在寻找对应样本点可以优劣势互补。因此下面我们考虑在DTW算法中使用BDS对应点匹配方法进行改进。

假设轨迹R的样本点个数为n，轨迹Q的样本点个数为m，并且我们已经通过DTW获得了所有DTW对应点对集合，我们记的DTW对应点集合为，并且由于允许一对多，所以可能包含轨迹Q的多个样本点，我们记其中时间序列最早的为，时间序列最晚的为。我们现在要基于BDS的思想，首先调整数据轨迹点的DTW对应点，称为的DTW-BDS对应点，简记为。并且记在轨迹段至间的BDS对应点为，且若和为同一个点，那么为。

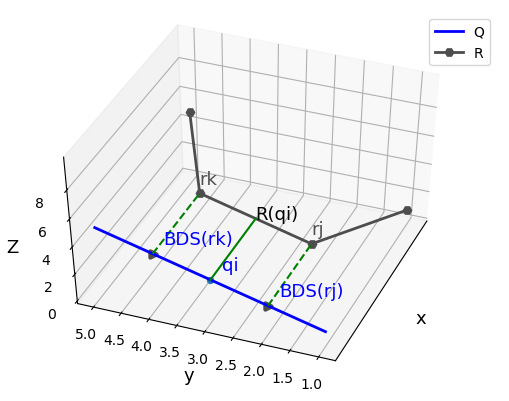
如果为轨迹R的第一个点，即的BDS对应点在至的DTW对应点中时间序列最早的点之间，即。如果恰好的第一个DTW对应点就是，那么。

如果不是轨迹R的第一个点，也不是最后一个点，且已经找到了其BDS对应点，则的对应点在的对应点与的对应点之间，因此。如图所示，为，为，那么在与之间。

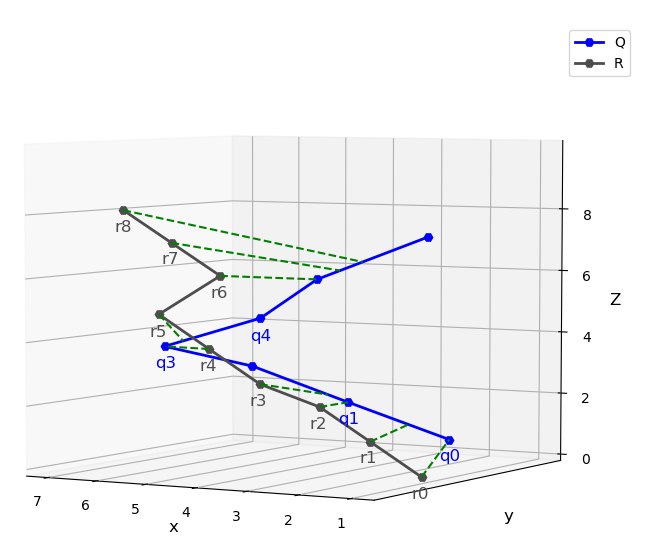


如果是轨迹R的最后一个点，那么。其中为轨迹Q的最后一个点。

在找到轨迹R上的所有样本点在Q上的对应点之后，我们再找轨迹Q在轨迹R上的对应点就简单了。如果想找在R上的对应点，通过时序关系，先找到的上一个BDS对应点和下一个BDS对应点，如图所示，那么的对应点在和之间，即。



在刚才的例子中，如果按照这种方法，先使用DTW算法获得轨迹R中每个样本点的DTW对应点，再使用BDS算法思想做局部优化，可以得到如下如所示的优化结果。我们发现轨迹R中的、、、和都分别找到了最佳的BDS对应点，而不再对应到轨迹Q中的样本点上，而、和的BDS对应点与DTW对应点相同，均为轨迹Q上的样本点。这样一来进一步减小了对应点之间的距离，因此该方法是行之有效的。



本文将这种先使用DTW算法寻找DTW对应点，再使用BDS算法中的思想进行局部优化的方法叫做DTW-BDS对应点匹配算法。由于本文是找出数据轨迹R中与查询轨迹Q最相似的部分，后面的算法中，只用到了数据轨迹段对应的查询轨迹段，因此这里只需要考虑数据轨迹R上的所有样本点在查询轨迹Q上的对应点即可，而不需要找出Q上的样本点在R上的对应点。因此这里不需要双向映射，而是单向映射，将数据轨迹R中的点向查询轨迹Q中映射。

本文后面使用到的对应点均为DTW-BDS算法寻找到的对应点，为方便起见，我们记样本点在轨迹Q上的DTW-BDS对应点为。

### 3.3.4对应轨迹段

在获得对应样本点的基础上，我们可以得到对应轨迹段的概念。数据轨迹R上的连续两个样本点和，会在查询轨迹Q上分别获得他们的对应点和，那么轨迹段的对应轨迹段就是。

对应轨迹段可能会出现多种情况，我们对每种情况的处理方式不尽相同。第一种情况是和之间的轨迹段是一条直线段。其中和有可能是轨迹Q的样本点，也有可能是轨迹Q中某两个样本点连线上的点。这个情况在讲对应点已经讨论过了，如图所示，的对应点是轨迹Q的样本点，的对应点在样本点和之间。

第二种情况是由于数据轨迹Q移动速度和移动距离等因素，和可能会对齐到Q上的同一个点，即和有可能是轨迹Q中的同一个样本点，如图所示，和的对应点和为同一个点，并且该点还是轨迹Q的样本点。

第三种情况是虽然和是两个连续的样本点，但是和中间可能隔着多个样本点，如图所示，和的对应点和中间夹着轨迹Q的样本点和。







由于后面计算轨迹相似性的需要，本文提出三维时空下的对应轨迹段相似性的概念。在三维时空中，数据轨迹段和对应的查询轨迹段之间的相似性由两部分组成，第一部分是其形状上的相似性，第二部分是轨迹段之间时空距离。下面依次对这两部分进行讲解。