DOI:10.13656/j.cnki.gxkx.2013.02.005

广西科学 Guangxi Sciences 2013,20(2):132～136

一种蛋白质三维结构中残基重心的计算方法*

An Algorithm of Calculating Protein Side Chain Centroids and Its Implementation with Python Programming Language

陆文伟[1**],韦廷宗[2],李　涛[3],杜丽琴[3],黄日波[2,3]
LU Wen-wei[1,2],WEI Ting-zong[2],LI Tao[3],DU Li-qin[3],HUANG Ri-bo[2,3]

(1.广西大学数学与信息科学学院,广西南宁　530004;2.广西科学院,广西南宁　530007;3.广西大学生命科学与技术学院,广西南宁　530004)
(1. Science and Technology College of Mathematics and Imformation Sciences, Guangxi University, Nanning, Guangxi, 530004, China; 2. Guangxi Academy of Sciences, Nanning, Guangxi, 530007, China; 3. College of Life Science and Biotechnology, Guangxi University, Nanning, Guangxi, 530004, China)

摘要:基于蛋白质三维结构数据 PDB,给出一种蛋白质残基重心的计算方法,同时阐述该方法的 python 语言的实现过程.该方法不需要知道残基侧链原子的坐标,只需知道主链上原子的坐标即可计算蛋白质残基间的距离.

关键词:PDB　球坐标　直角坐标　残基重心　python 语言
中图法分类号:TP312,Q816　　**文献标识码**:A　　**文章编号**:1005-9164(2013)02-0132-05

Abstract: In some applications such as protein structure prediction or molecular docking, distances between pairs of residues play an important role. Actually, this distances are those between pairs of residue centroids. This paper describes in detail an algorithm of how to calculate the residue centroids and its implement with python programming language.

Key words: PDB, spherical coordinates, cartesian coordinates, residue centroid, python programming language

在蛋白质统计势能的计算[1],分子力学力场(force field),以及分子对接等应用中都有可能要计算蛋白质残基间的距离.很多时候,为了节约计算资源,蛋白质的三维结构通常使用一个粗粒度模型,即简化的蛋白质结构模型表示[2].在这个简化模型中,蛋白质残基侧链通常由一个伪原子来表示,这个伪原子相当于蛋白质侧链的重心.蛋白质残基间的距离即指残基重心间的距离.因此计算残基间的距离,必须事先计算残基的重心.

PDB 是常用的蛋白质三维结构数据[3],本文主要基于这种数据给出一种蛋白质残基重心的计算方法,并阐述该方法的 python 语言实现过程.

1　蛋白质残基重心计算方法

一般情况下,假设某个残基侧链有 n 个重原子,第 i 个重原子的三维坐标为 (x_i, y_i, z_i),则残基重心(几何重心)的坐标为

$$(x, y, z) = (\frac{1}{n}\sum_{i=1}^{n} x_i, \frac{1}{n}\sum_{i=1}^{n} y_i, \frac{1}{n}\sum_{i=1}^{n} z_i).$$

有一种常用的残基重心的计算方法,是将残基重心看成一个伪原子 C_μ,而这个伪原子通过一个虚拟的键和主链上的 C_α 原子相连[4](图 1).该方法在一个构造良好的蛋白质结构数据库中,计算各种残基的所有侧链重原子的几何重心后,再通过统计平均得出几何重心平均值.把伪原子 C_μ 的位置用球坐标 $\{r, \theta, \varphi\}$ 表示(图 2),这个球坐标参考了残基在主链上的三个

收稿日期:2012-10-11
修回日期:2012-10-26
作者简介:陆文伟(1970-),男,副研究员,主要从事蛋白质工程研究。
*国家 973 项目(2009CB724703);国家自然科学基金项目(31060125)资助。
**通讯作者:陆文伟,lu_wenwei@163.com;luwenwei@gxu.edu.cn

重原子（N,C_α,C）的几何位置：原子 C_α 在坐标原点上，原子 N 在 z 轴上，原子 C 在平面 $z-x$ 上，靠近 x 轴[5]. 侧链重心的球坐标来源于某个蛋白质数据库所有该残基侧链重原子几何重心的平均值，但是残基种类不同该数值会有所不同. 这种处理方法有一个好处，即残基重心的坐标值与侧链结构无关，只需知道主链原子的 PDB 坐标值，就可以通过坐标转换求出残基重心在 PDB 坐标系里的位置.

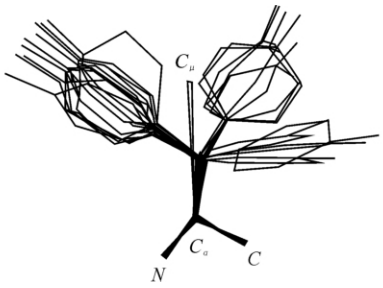

图 1　残基伪原子 C_μ 位置的确定

Fig. 1　Definition of the centroid positions C_μ

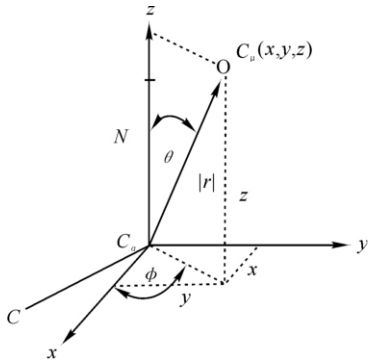

图 2　球坐标 C_μ 及其直角坐标的关系

Fig. 2　Spherical coordinates and Cartesian ones

我们利用表 1 中的数据[4]：其中，第 1 列是残基的单字母表示，第 2 列是原子 C_α 和 C_μ 的距离，第 3 列是原子 $N-C_\alpha-C_\mu$ 所成角的大小，第 4 列是二面角 $C-N-C_\alpha-C_\mu$ 的值，以二面角 $C-N-C_\alpha-C_\beta$ 为参照（注意，甘氨酸无需转换，其侧链重心就是 C_α 原子），将 C_μ 原子的球坐标转换为以 C_α 为原点的直角坐标，再进一步转换为 PDB 所使用的坐标，将所有残基的重心 C_μ 统一到 PDB 的坐标上来，这样就能计算蛋白质残基间的距离.

根据球坐标和直角坐标的转换定理，球坐标 $\{r,\theta,\varphi\}$ 可依据下面的公式转换成直角坐标 (x,y,z)，转换关系如图 3，其中点 B 表示 C_μ 原子，点 O 表示 C_α 原子，点 M 为 C_μ 在 $x-y$ 平面上的投影，α 表示 $\angle CON$，θ 表示 $\angle NOB$，φ 表示 $\angle MOL$，也就是二面角 $C-N-C_\alpha-C_\mu$（是平面 CNOL 和平面 MONB 所夹的角）. 另外，假设 $|CO|=c$，$|BO|=r$，$|NO|=h$.

$$\begin{cases} x = r\sin\theta\cos\varphi, \\ y = r\sin\theta\sin\varphi, r \geqslant 0, 0 \leqslant \theta \leqslant \pi, \\ \quad 0 \leqslant \varphi < 2\pi, \\ z = r\cos\theta. \end{cases} \tag{1}$$

表 1　20 种残基重心平均球坐标

Table 1　Average spherical coordinates of 20 residue centrolds

残基	r	θ	φ
A	0.764	110.103	0.0
C	1.375	108.455	10.221
D	1.984	110.414	12.862
E	2.624	110.497	16.332
F	2.985	113.852	15.607
G	0.0	0.0	0.0
H	2.708	110.676	17.050
I	1.862	110.297	11.269
K	2.940	112.861	16.354
L	2.088	113.633	20.670
M	2.052	113.248	17.990
N	1.979	110.759	14.459
P	1.409	64.545	−1.752
Q	2.593	112.027	19.167
R	3.631	113.305	17.432
S	1.259	108.115	−0.922
T	1.460	109.261	4.311
V	1.474	113.770	6.826
W	3.500	117.390	14.186
Y	3.359	112.396	15.749

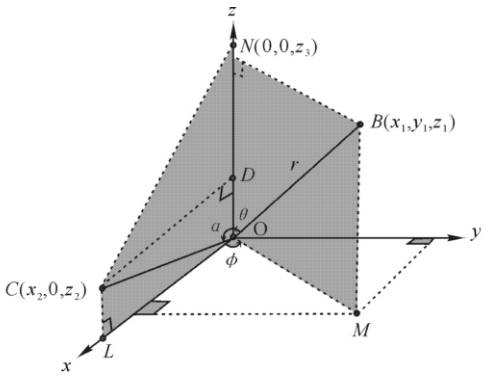

图 3　坐标转换

Fig. 3　The transformation of spherical coordinates of C_μ to Cartesian ones

由于球坐标的特殊取向，其原点在 C_α 原子上，因此 C 原子、C_α 和 N 的直角坐标可表示为：$C(x_2,0,z_2)$，$C_\alpha(0,0,0)$，$N(0,0,z_3)$. x_2 和 z_2 可依据边长 c 和角 α 求得，即 $x_2=c\sin\alpha$，$z_2=c\cos\alpha$，z_3 即为边长 h. 边长 c 是原子 C 和 C_α 间的键长，可由这两个原子的 PDB 坐标求得，角 α 则由三角形的余弦定理求得：设边长 $|CN|=a$，由原子 C 和 N 的 PBD 坐标确定，因此有 $\cos\alpha=\dfrac{h^2+c^2-a^2}{2hc}$. 假设 C_μ 原子的直角坐标为 (x_1,y_1,z_1)，由于残基的球坐标已知，故其笛卡

儿坐标可根据转换公式(1)求得.

得到原子在以 C_α 为原点的笛卡儿坐标后,再通过直角坐标的旋转变换(图 4)得到. 与 PDB 坐标同型的坐标系(即坐标轴指向相同而原点不同的坐标系).

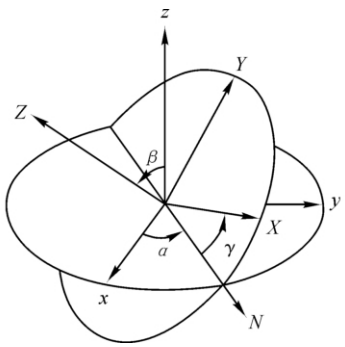

图 4　用于坐标旋转转换关系的欧拉角(α,β,γ)

Fig. 4　Euler angles-the xyz (fixed) system

xyz 表示原坐标, XYZ 表示旋转后坐标.

The XYZ (rotated) system and the line of nodes, labeled N.

坐标的旋转涉及到具有相同原点的两个直角坐标,可用一个三元组的欧拉角 (α,β,γ) 描述. 根据 x, y, z 三个坐标轴旋转先后的不同,可以有多种旋转方法,例如命名 (YXZ) 表示先转 Y 轴,而后转 X, Z 两轴. 在这里采用 (YXZ) 方法,通过下述旋转矩阵实现:

$$R=\begin{pmatrix}\cos\alpha\cos\gamma+\sin\alpha\sin\beta\sin\gamma & \cos\gamma\sin\alpha\sin\beta-\cos\alpha\sin\gamma & \cos\beta\sin\alpha\\ \cos\beta\sin\gamma & \cos\beta\cos\gamma & -\sin\beta\\ \cos\alpha\sin\beta\sin\gamma-\cos\gamma\sin\alpha & \sin\alpha\sin\gamma+\cos\alpha\cos\gamma\sin\beta & \cos\alpha\cos\beta\end{pmatrix}. \tag{2}$$

如果以球坐标下的直角坐标做为参照系,那么和 PDB 坐标系同型的导出坐标就可以用下面的方法求出:令 (x_r,y_r,z_r) 为参照坐标, (x_d,y_d,z_d) 为导出坐标,则有

$$R\begin{bmatrix}x_r\\y_r\\z_r\end{bmatrix}=\begin{bmatrix}x_d\\y_d\\z_d\end{bmatrix}. \tag{3}$$

$C(x_2,0,z_2)$, $C_\alpha(0,0,0)$, $N(0,0,z_3)$ 都是参照坐标,导出坐标则是原子 C, C_α, N 的 PDB 坐标平移到 C_α 原子后所得的坐标. 把他们分别代入上式,即可求出 (α,β,γ),这样就可以利用公式(2)求出 C_μ 的导出坐标. 再平移回 PDB 坐标系即可得到其真正的 PDB 坐标.

根据公式(3),对原子 $N(0,0,z_3)$ 而言,由如下公式求得 α, β:

$$\begin{cases}z_3\cos\beta\sin\alpha=x_d,\\-z_3\sin\beta=y_d,\\z_3\cos\alpha\cos\beta=z_d.\end{cases}\Rightarrow\begin{cases}\sin\beta=-\dfrac{y_d}{z_3},\\ \\ \mathrm{tg}\alpha=\dfrac{x_d}{z_d}.\end{cases} \tag{4}$$

对原子 $C(x_2,0,z_2)$ 而言,由如下公式求得 γ:

$$x_2\cos\beta\sin\gamma-z_2\sin\beta=y_d\Rightarrow\sin\gamma=\frac{y_d+z_2\sin\beta}{x_2\cos\beta}. \tag{5}$$

从上面的分析可以看出,原子 C, C_α, N 的 PDB 坐标都是做为辅助工具以求取 C_μ 的 PDB 坐标值. 通过它们可以求得主链原子间的距离和所成的角,由此可求得旋转矩阵和平移距离,最终解出伪原子的 PDB 坐标值.

2　残基重心计算方法的 python 语言实现

根据已知的各种残基重心的平均球坐标和特定蛋白质 PDB 中主链原子的坐标计算残基重心,包括以下步骤:

1)计算残基在主链上的 3 个重原子在球坐标下的直角坐标;

2)计算残基重心在球坐标下的直角坐标;

3)计算残基重心的直角坐标,这个坐标和 PDB 坐标同型,但原点不同;

4)平移步骤 3)所得的坐标,使其和 PDB 坐标一致.

蛋白质残基重心的 python 语言[6]实现过程如下:

先定义一些相关的数据结构和函数. 定义一个字典数据类型(dictionary),其键(key)为残基的三字母或单字母代码,键值(value)则为残基的平均球坐标,用一个三元组(tuple)表示,如对丙氨酸残基而言,有:

avg_centroid[" A"] = (0.764,110.103,0.0).

avg_centroid 是字典名称,赋值号右边的元组表示球坐标,其余氨基酸残基如此类推.

定义一个矢量类 Vector3d,负责处理矢量的加减、内积和外积等的计算.

定义函数 reference_xyz,负责计算参考坐标,即残基主链原子 C, C_α, N 相应于球坐标系的直角坐标值 $C(x_2,0,z_2)$, $C_\alpha(0,0,0)$, $N(0,0,z_3)$:

```
def reference_xyz(v_c_pdb, v_ca_pdb, v_n_pdb):
    """"""
    distance_n_ca = vector3d.pos_distance(v_ca_pdb, v_n_pdb)
```

```
        distance_n_c = vector3d.pos_distance(v_c_pdb,v_n_pdb)
        distance_c_ca = vector3d.pos_distance(v_ca_pdb,v_c_pdb)
        m=distance_c_ca ** 2 + distance_n_ca ** 2-distance_n_c ** 2
        n=2 * distance_c_ca * distance_n_ca
        a=math.acos(m / n)
        x2 = distance_c_ca * math.sin(a)
        z2 = distance_c_ca * math.cos(a)
        z3 = distance_n_ca
        v_c = vector3d.Vector3d(x2,0,z2)
        v_n = vector3d.Vector3d(0,0,z3)
        return (v_c,v_n)
```

函数参数 v_c_pdb,v_ca_pdb,v_n_pdb **分别表示主链原子在** PDB **中的坐标. 函数返回原子** C **和** N **在球坐标下的直角坐标.**

定义函数 sphere2xyz,**负责将球坐标转换为直角坐标,由此可得到残基重心** C_μ **相应于球坐标的直角坐标值** (x_1,y_1,z_1):

```
def sphere2xyz(distance,angle,dihedral):
    """"""
        x = distance * math.sin(angle) * math.cos(dihedral)
        y = distance * math.sin(angle) * math.sin(dihedral)
        z = distance * math.cos(angle)
        v = vector3d.Vector3d (x,y,z)
        return v
```

函数参数 distance,angle,dihedral **分别表示残基重心的球坐标,返回值是表示残基重心直角坐标的矢量.**

定义函数 eulerAngle,**负责计算坐标转换用到的欧拉角** (α,β,γ):

```
def eulerAngle(v_c_reference,v_n_reference,v_c_pdb,v_n_pdb,v_ca_pdb):
    """"""
        v_n = v_n_pdb - v_ca_pdb
        v_c = v_c_pdb - v_ca_pdb
        if v_n_reference.z == 0.0 or v_n_reference.z < SMALL:
            b = math.asin(0.0)
        else:
            b = math.asin(-v_n.y / v_n_reference.z)
        if v_n.z == 0.0 or v_n.z < SMALL:
            a = math.atan(0.0)
        else:
            a = math.atan(v_n.x / v_n.z)
        m = v_c.y + math.sin(b) * v_c_reference.z
        n = math.cos(b) * v_c_reference.x
        if n == 0.0 or n < SMALL:
            r = math.acos(0.0)
        elif m - n < SMALL:
            r = math.acos(1.0)
        else:
            r = math.acos(m/n)
        return (a,b,r)
```

函数参数 v_c_reference,v_n_reference,v_c_pdb,v_n_pdb,v_ca_pdb **分别表示** C **和** N **原子在球坐标下的直角坐标,以及** C、N **和** C_α **原子在** PDB **中的坐标. 函数返回欧拉角,用元组**(a,b,r)**表示.**

代码中变量 SMALL=1E-6,**其字面值用科学标记方式表示,表示在浮点计算中,当变量的值小于这个数时,变量的值当作零看待.**

定义函数 derive_xyz,**负责计算残基重心** C_μ **的导出坐标,即以原子** C_α **为原点,且和** PDB **数据同型的直角坐标:**

```
def derive_xyz(a,b,r,v_cu):
    """"""
        sin_a = math.sin(a)
        cos_a = math.cos(a)
        sin_b = math.sin(b)
        cos_b = math.cos(b)
        sin_r = math.sin(r)
        cos_r = math.cos(r)
        a11 = cos_a * cos_r + sin_a * sin_b * sin_r
        a12 = cos_r * sin_a * sin_b - cos_a * sin_r
        a13 = cos_b * sin_a
        a21 = cos_b * sin_r
        a22 = cos_b * cos_r
        a23 = -sin_b
        a31 = cos_a * sin_b * sin_r - cos_r * sin_a
        a32 = sin_a * sin_r + cos_a * cos_r * sin_b
        a33 = cos_a * cos_b
```

```
        x = a11 * v_cu.x + a12 * v_cu.y + a13 * v_cu.z
        y = a21 * v_cu.x + a22 * v_cu.y + a23 * v_cu.z
        z = a31 * v_cu.x + a32 * v_cu.y + a33 * v_cu.z
        centroid = vector3d.Vector3d(x,y,z)
        return centroid
```

函数参数 a,b,r,v_cu 分别表示欧拉角(a,b,r)和残基重心在球坐标下的直角坐标.函数返回残基重心的直角坐标,这个坐标和 PDB 坐标同型,但原点不同.

将所得的残基重心坐标 centroid 经坐标平移后即可与 PDB 数据的坐标一致.另外,代码中使用到 python 的常用的数学函数标准库 math,在源代码的最开始处必须先导入:import math.

本文只给出蛋白质残基重心计算方法在 python 语言下的实现过程,还可以有其他的实现方法,甚至于不同的编程语言,如 C、C++等.一般来讲,我们可根据不同的要求进行选择.

参考文献:

[1] Dehouck Y, Gilis D, Rooman M. A new generation of statistical potentials for proteins[J]. Biophys J, 2006, 90(11): 4010-7.

[2] Levitt M. A simplified representation of protein conformations for rapid simulation of protein folding[J]. J Mol Biol, 1976, 104(1): 59-107.

[3] Bernstein F C, Koetzle TF, Williams GJB, et al. The Protein Data Bank: a computer-based archival file for macromolecular structures[J]. J Mol Biol, 1977, 112(3): 535-42.

[4] Kocher J P A, Rooman M J, Wodak S J. Factors influencing the ability of knowledge-based potentials to identify native sequence-structure matches[J]. Journal of Molecular Biology, 1994, 235(5): 1598-1613.

[5] Betancourt M R. A reduced protein model with accurate native-structure identification ability [J]. Proteins: Structure, Function and Genetics, 2003, 53(4): 889-907.

[6] Lutz M. Programming python[M]. 4th Edition. O'Reilly Media, 2010.

(责任编辑:尹 闯)

(上接第 131 页 Continue from page 131)

To sum up, BP network based on rough set genetic algorithm has high accuracy identification, its speed is fast, and generalization ability is strong, therefore, it is a promising classification method.

References:

[1] Zhang J M, Song Y Q, Zhou S W, et al. Survey on data aggregation techniques in wireless sensor networks[J]. Computer Applications, 2006, 26(6): 1273-1283.

[2] Yuan J B, Pu H C. E-mail information classifier of neural network based on gennetic algorithm Optimization[J]. Nanjing University of Science and Technology: Natural Science, 2008, 32(1): 78-81.

[3] Li S, Liu L J, Xie Y L. Chaotic prediction for short-term traffic flow of optimized BP neural network based on genetic algorithm[J]. Control and Decision, 2011, 26(10): 1581-1585.

[4] Wang Z W. Improved BP neural network-based license plate character recognition [J]. Microelectronics and Computer, 2011, 28(9): 66-69.

[5] Wang H T, Huang Z H, Fang X G, et al. The application of rough sets - neural network theory to mine ventilation system evaluation[J]. Chongqing University, 2011, 34(9): 90-94.

[6] Hong Y H. Stdudy on distributed data mining algorithm in wireless sensor networks[J]. Computer Simulation, 2012, 29(12): 167-170.

[7] Nie Y, Chen F J. Research of K-mean clustering algorithm based on rough set[J]. Engineering Journal of Wuhan University, 2011, 44(2): 257-260.

[8] Zheng W, Zhou Z Q, Ma Y L. A feature selection approach based on rough set theory [J]. Hebei North University: Natural Science Edition, 2009, 5(1): 56-59.

[9] Luo G Z, Yang X J. Rough attributes reduction algorithm based on partitioning strategy merge[J]. Statistics and Decision, 2009(20): 146-148.

[10] Cao L. Research on the colliery safety wireless monitoring system basedon multisource information fusion teehnique[D]. Xian: Xian University of Architecture and Technology, 2008.

(责任编辑:尹 闯)