

第七讲作业

崔华坤

二、Bundle Adjustment

2.1 文献阅读

1. 为何说 BA is slow 不对？

答：因为没有考虑到 J 和 H 矩阵的稀疏性，即只存在位姿与路标点的相互关系，而位姿与位姿之间，及路标点与路标点之前没有约束关系，因此对应的矩阵块是对角矩阵，这样可以先进行边缘化，将矩阵求解分解成先求只有位姿的矩阵方程，再求路标点方程。

2. BA 中需要注意参数化的地方有：3D 路标点、旋转。

3D 路标点可以表示成非齐次和齐次，若用非齐次表示，当点距离很远时，此时的 X,Y,Z 值都很大，此时的代价函数会变得很单调，步长将变得很大；若用齐次表示，则可令 $W=0$ ，这里没有理解齐次的作用？

旋转的参数化方式：可以用欧拉角，但存在约束和万向锁问题；可以用四元数；也可以用旋转矩阵，进行局部扰动。

3. H 的稀疏性可以实现 BA 实时，在 07 年的 PTAM 上实现。

2.2 BAL-dataset

误差项为：

$$e(\xi, f, k_1, k_2, P_w) = P_{uv} - K P_{cd} = P_{uv} - K r(P_{cn}) \quad (1)$$

$$\text{其中, } P_c = \exp(\xi^\wedge) P_w$$

其中， P_{uv} 为像素坐标系， K 为相机内参矩阵， $r(\cdot)$ 为畸变校正变换， P_{cd} 为畸变校正后的归一化的相机坐标系， P_{cn} 为归一化的相机坐标系， P_c 为相机坐标系， P_w 为世界坐标系。

值得说明的是，这里的整体 Jacobian 为 2×12 ，为：

$$J = [J_{\delta\xi}^{2 \times 6}, J_f^{2 \times 1}, J_{k_1}^{2 \times 1}, J_{k_2}^{2 \times 1}, J_{P_w}^{2 \times 3}]$$

下面我们分别讨论：

1. 对于第一项 $J_{\delta\xi}^{2 \times 6}$ ，根据链式法则：

$$\frac{\partial e}{\partial \delta \xi} = \frac{\partial e}{\partial P_c} \cdot \frac{\partial P_c}{\partial \delta \xi} \quad (2)$$

将(1)式详细展开：

$$e = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} - \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -\frac{x_c}{z_c}(1 + k_1 r^2 + k_2 r^4) \\ -\frac{y_c}{z_c}(1 + k_1 r^2 + k_2 r^4) \\ 1 \end{bmatrix} \quad (3)$$

可化简为：

$$e_x = u + \frac{f x_c}{z_c} (1 + k_1 r^2 + k_2 r^4) \quad (4)$$

$$e_y = v + \frac{f y_c}{z_c} (1 + k_1 r^2 + k_2 r^4)$$

其中， $r^2 = \left(\frac{x_c}{z_c}\right)^2 + \left(\frac{y_c}{z_c}\right)^2$ ， $c_x = c_y = 0$

注意 BAL 数据集集中的从相机坐标系到归一化的投影模型多了一个负号，这是因为假设的像平面不在相机光心和 3D 物体之间，而是在另外一侧。

对(4)进行求导：

$$\frac{\partial e_x}{\partial x_c} = \frac{f}{z_c} (1 + k_1 r^2 + k_2 r^4) + \frac{2f x_c^2}{z_c^3} (k_1 + 2k_2 r^2)$$

$$\frac{\partial e_x}{\partial y_c} = \frac{2f x_c y_c}{z_c^3} (k_1 + 2k_2 r^2)$$

$$\frac{\partial e_x}{\partial z_c} = -\frac{f x_c}{z_c^2} (1 + k_1 r^2 + k_2 r^4) - \frac{2f x_c r^2}{z_c^2} (k_1 + 2k_2 r^2)$$

$$\frac{\partial e_y}{\partial x_c} = \frac{2f x_c y_c}{z_c^3} (k_1 + 2k_2 r^2)$$

$$\frac{\partial e_y}{\partial y_c} = \frac{f}{z_c} (1 + k_1 r^2 + k_2 r^4) + \frac{2f y_c^2}{z_c^3} (k_1 + 2k_2 r^2)$$

$$\frac{\partial e_y}{\partial z_c} = -\frac{f y_c}{z_c^2} (1 + k_1 r^2 + k_2 r^4) - \frac{2f y_c r^2}{z_c^2} (k_1 + 2k_2 r^2)$$

对于公式(2)的第二部分：参考《十四讲》4.3.5

$$\frac{\partial P_c}{\partial \delta \xi} = [I \quad -P_c^\wedge]$$

注意 $\frac{\partial P_c}{\partial \delta \xi}$ 若 se3 的旋转在前，平移在后，需要将前三列和后三列互换。

2. 下面讨论 $J_f^{2 \times 1}$ ：

根据公式(4)可计算出：

$$\frac{\partial e}{\partial f} = \begin{bmatrix} \frac{x_c}{z_c} (1 + k_1 r^2 + k_2 r^4) \\ \frac{y_c}{z_c} (1 + k_1 r^2 + k_2 r^4) \end{bmatrix}$$

3. 下面讨论 $J_{k_1}^{2 \times 1}, J_{k_2}^{2 \times 1}$ ：

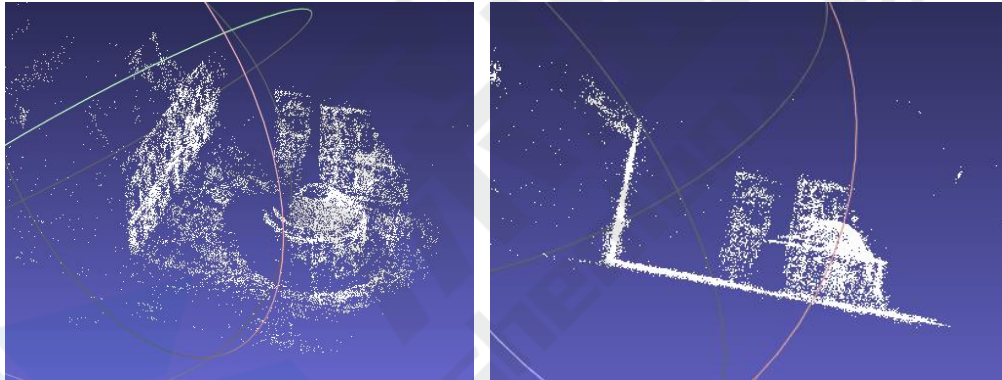
根据公式(4)可计算出：

$$\frac{\partial e}{\partial k} = \begin{bmatrix} \frac{f x_c}{z_c} r^2 & \frac{f x_c}{z_c} r^4 \\ \frac{f y_c}{z_c} r^2 & \frac{f y_c}{z_c} r^4 \end{bmatrix}$$

4. 下面讨论 $J_{P_w}^{2 \times 3}$ ：参考《十四讲》7.47

$$\frac{\partial e}{\partial P_c} \cdot \frac{\partial P_c}{\partial P_w} = \frac{\partial e}{\partial P_c} R$$

因其他 BAL 数据显示有问题，点都堆积在一起，但是下面数据没问题，因此仍是用下面数据测试：



三、直接法的 Bundle Adjustment

3.1 数学模型

1. 任意一点 P_w 投影在任意一图像中的误差项可写为：

$$e(P_w, \xi) = I(P_w) - I_i \left(\frac{1}{z_c} K \exp(\xi^\wedge) P_w \right) \quad (5)$$

其中，第一项通过 3D 点集 points.txt 获得，第二项需要根据 16 个像素的位置关系，计算不同位置的灰度值。

2. 每个误差项 error 关联两个优化变量，分别为： P_w, ξ 。

3. error 关于各变量的雅可比矩阵为：

$$J_{P_w} = \frac{\partial e}{\partial P_w} = \frac{\partial [I(P_w) - I_i(\frac{1}{z_c} K \exp(\xi^\wedge) P_w)]}{\partial P_w} \quad (6)$$

令：

$$P_{uv} = \frac{1}{z_c} K \exp(\xi^\wedge) P_w$$

$$P_c = \exp(\xi^\wedge) P_w$$

则(6)式可整理为：

$$J_{P_w} = - \frac{\partial I_i}{\partial P_{uv}} \frac{\partial P_{uv}}{\partial P_c} \frac{\partial P_c}{\partial P_w}$$

其中，

$$\frac{\partial I_i}{\partial P_{uv}} = \left[\frac{(I_i(u+1, v) - I_i(u-1, v))/2}{(I_i(u, v+1) - I_i(u, v-1))/2} \right]$$

$$\frac{\partial P_{uv}}{\partial P_c} = \begin{bmatrix} \frac{f_x}{z_c} & 0 & -\frac{f_x x_c}{z_c^2} \\ 0 & \frac{f_y}{z_c} & -\frac{f_y y_c}{z_c^2} \end{bmatrix}$$

$$\frac{\partial P_c}{\partial P_w} = R$$

第二项 Jacobian 为：

$$J_{\delta \xi} = \frac{\partial e}{\partial \delta \xi} = \lim_{\delta \xi \rightarrow 0} \frac{e(\xi \oplus \delta \xi) - e}{\delta \xi}$$

其中，

$$e(\xi \oplus \delta \xi) - e = I(P_w) - I_i \left(\frac{1}{z_c} K \exp(\delta \xi^\wedge) \exp(\xi^\wedge) P_w \right) - e$$

$$= I(P_w) - I_i \left(\frac{1}{z_c} K \exp(\xi^\wedge) P_w + \frac{1}{z_c} K \delta \xi^\wedge \exp(\xi^\wedge) P_w \right) - e \quad (7)$$

同样，令：

$$P_{uv} = \frac{1}{z_c} K \exp(\xi^\wedge) P_w$$

$$P_c = \exp(\xi^\wedge) P_w$$

则，(7)式可整理为：

$$e(\xi \oplus \delta \xi) - e = - \frac{\partial I_i}{\partial P_{uv}} \frac{\partial P_{uv}}{\partial P_c} \frac{\partial P_c}{\partial \delta \xi} \delta \xi$$

则：

$$J_{\delta \xi} = \lim_{\delta \xi \rightarrow 0} \frac{e(\xi \oplus \delta \xi) - e}{\delta \xi} = - \frac{\partial I_i}{\partial P_{uv}} \frac{\partial P_{uv}}{\partial P_c} \frac{\partial P_c}{\partial \delta \xi}$$

其中，前两项上面已给出，最后一项为：

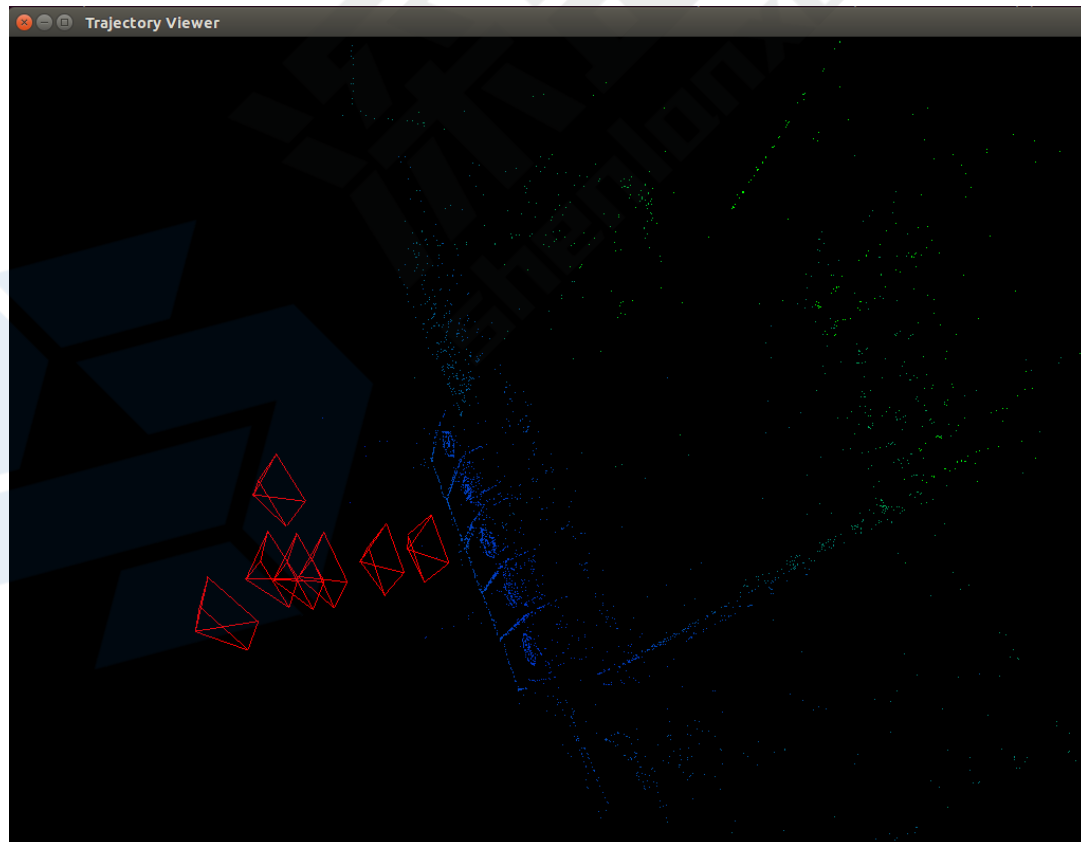
$$\frac{\partial P_c}{\partial \delta \xi} = [I \quad -P_c^A]$$

3.2 实现

```

cui@ubuntu: ~/Project/SLAMCourse/IT-3-direct-ba2/build
Iterations: 2   ch12= 3241145.581652   time= 0.126546   cunTime= 0.386247   edges= 28826   schur= 1   lambda= 41151.378708   levenbergIter= 1
Iterations: 3   ch12= 3174691.646463   time= 0.124514   cunTime= 0.510761   edges= 28826   schur= 1   lambda= 13717.126236   levenbergIter= 1
Iterations: 4   ch12= 3122770.671499   time= 0.126593   cunTime= 0.637265   edges= 28826   schur= 1   lambda= 4572.375412   levenbergIter= 1
Iterations: 5   ch12= 3087767.342058   time= 0.124854   cunTime= 0.762118   edges= 28826   schur= 1   lambda= 3848.258275   levenbergIter= 1
Iterations: 6   ch12= 3058711.784658   time= 0.124444   cunTime= 0.886562   edges= 28826   schur= 1   lambda= 2832.166850   levenbergIter= 1
Iterations: 7   ch12= 3025310.897499   time= 0.124146   cunTime= 1.01071   edges= 28826   schur= 1   lambda= 1354.777900   levenbergIter= 1
Iterations: 8   ch12= 3007317.292265   time= 0.124011   cunTime= 1.13472   edges= 28826   schur= 1   lambda= 903.185267   levenbergIter= 1
Iterations: 9   ch12= 2994855.419457   time= 0.123774   cunTime= 1.25849   edges= 28826   schur= 1   lambda= 602.123511   levenbergIter= 1
Iterations: 10  ch12= 2978774.612668   time= 0.125001   cunTime= 1.38349   edges= 28826   schur= 1   lambda= 401.415674   levenbergIter= 1
Iterations: 11  ch12= 2970880.128675   time= 0.1256   cunTime= 1.50989   edges= 28826   schur= 1   lambda= 267.618449   levenbergIter= 1
Iterations: 12  ch12= 2963939.983889   time= 0.125446   cunTime= 1.63454   edges= 28826   schur= 1   lambda= 178.406966   levenbergIter= 1
Iterations: 13  ch12= 2961224.546037   time= 0.12454   cunTime= 1.75908   edges= 28826   schur= 1   lambda= 118.937977   levenbergIter= 1
Iterations: 14  ch12= 2957345.182912   time= 0.123072   cunTime= 1.88215   edges= 28826   schur= 1   lambda= 79.291985   levenbergIter= 1
Iterations: 15  ch12= 2954168.709604   time= 0.124472   cunTime= 2.00663   edges= 28826   schur= 1   lambda= 52.861323   levenbergIter= 1
Iterations: 16  ch12= 2952718.499025   time= 0.137877   cunTime= 2.1445   edges= 28826   schur= 1   lambda= 35.240882   levenbergIter= 1
Iterations: 17  ch12= 2948849.571016   time= 0.142242   cunTime= 2.28674   edges= 28826   schur= 1   lambda= 23.493921   levenbergIter= 1
Iterations: 18  ch12= 2946491.320899   time= 0.13289   cunTime= 2.41963   edges= 28826   schur= 1   lambda= 15.662614   levenbergIter= 1
Iterations: 19  ch12= 2945190.319115   time= 0.218699   cunTime= 2.63833   edges= 28826   schur= 1   lambda= 668.271544   levenbergIter= 4
Iterations: 20  ch12= 2942656.832981   time= 0.143409   cunTime= 2.78174   edges= 28826   schur= 1   lambda= 445.514363   levenbergIter= 1
Iterations: 21  ch12= 2941947.584215   time= 0.16226   cunTime= 2.944   edges= 28826   schur= 1   lambda= 594.019130   levenbergIter= 2
Iterations: 22  ch12= 2939876.495167   time= 0.181286   cunTime= 3.12529   edges= 28826   schur= 1   lambda= 3168.102136   levenbergIter= 3
Iterations: 23  ch12= 2937695.165161   time= 0.129033   cunTime= 3.25432   edges= 28826   schur= 1   lambda= 2112.068091   levenbergIter= 1
Iterations: 24  ch12= 2937127.904622   time= 0.154886   cunTime= 3.40921   edges= 28826   schur= 1   lambda= 2816.090788   levenbergIter= 2
Iterations: 25  ch12= 2935991.344962   time= 0.156528   cunTime= 3.56574   edges= 28826   schur= 1   lambda= 3754.787717   levenbergIter= 2
Iterations: 26  ch12= 2934138.600702   time= 0.133532   cunTime= 3.69927   edges= 28826   schur= 1   lambda= 2503.191811   levenbergIter= 1
Iterations: 27  ch12= 2925842.473822   time= 0.196846   cunTime= 3.89611   edges= 28826   schur= 1   lambda= 106582.858689   levenbergIter= 4
Iterations: 28  ch12= 2922550.173721   time= 0.124271   cunTime= 4.02038   edges= 28826   schur= 1   lambda= 71201.980406   levenbergIter= 1
Iterations: 29  ch12= 2918593.158963   time= 0.168082   cunTime= 4.18847   edges= 28826   schur= 1   lambda= 379743.468833   levenbergIter= 3
Iterations: 30  ch12= 2916660.112728   time= 0.124524   cunTime= 4.31299   edges= 28826   schur= 1   lambda= 253162.312556   levenbergIter= 1
Iterations: 31  ch12= 2916554.676474   time= 0.145798   cunTime= 4.45879   edges= 28826   schur= 1   lambda= 337549.750074   levenbergIter= 2
Iterations: 32  ch12= 2915767.094682   time= 0.123961   cunTime= 4.58275   edges= 28826   schur= 1   lambda= 225033.166716   levenbergIter= 1
Iterations: 33  ch12= 2913880.990236   time= 0.167623   cunTime= 4.75037   edges= 28826   schur= 1   lambda= 1280176.889152   levenbergIter= 3
Iterations: 34  ch12= 2913406.579218   time= 0.123962   cunTime= 4.87434   edges= 28826   schur= 1   lambda= 808117.926181   levenbergIter= 1
Iterations: 35  ch12= 2913365.486351   time= 0.144937   cunTime= 5.01927   edges= 28826   schur= 1   lambda= 1066823.901469   levenbergIter= 2
Iterations: 36  ch12= 2913189.688596   time= 0.145457   cunTime= 5.16473   edges= 28826   schur= 1   lambda= 1422431.868625   levenbergIter= 2
Iterations: 37  ch12= 2913143.036744   time= 0.12314   cunTime= 5.28787   edges= 28826   schur= 1   lambda= 948287.912417   levenbergIter= 1
Iterations: 38  ch12= 2912973.408068   time= 0.14542   cunTime= 5.43329   edges= 28826   schur= 1   lambda= 1264383.883222   levenbergIter= 2
Iterations: 39  ch12= 2912593.851483   time= 0.167536   cunTime= 5.60083   edges= 28826   schur= 1   lambda= 6743380.710519   levenbergIter= 3
Iterations: 40  ch12= 2912505.191385   time= 0.125411   cunTime= 5.72624   edges= 28826   schur= 1   lambda= 4495587.140345   levenbergIter= 1
Iterations: 41  ch12= 2912433.588049   time= 0.167423   cunTime= 5.89366   edges= 28826   schur= 1   lambda= 2397464.748507   levenbergIter= 3
Iterations: 42  ch12= 2912429.534803   time= 0.146535   cunTime= 6.04019   edges= 28826   schur= 1   lambda= 31968619.664676   levenbergIter= 2
Iterations: 43  ch12= 2912429.129300   time= 0.16793   cunTime= 6.20812   edges= 28826   schur= 1   lambda= 170499304.878274   levenbergIter= 3
Iterations: 44  ch12= 2912428.899513   time= 0.125441   cunTime= 6.33356   edges= 28826   schur= 1   lambda= 113666203.252183   levenbergIter= 1
Iterations: 45  ch12= 2912428.732847   time= 0.168728   cunTime= 6.50229   edges= 28826   schur= 1   lambda= 606219750.678307   levenbergIter= 3
Iterations: 46  ch12= 2912428.725055   time= 0.145677   cunTime= 6.64797   edges= 28826   schur= 1   lambda= 808293000.904410   levenbergIter= 2
Iterations: 47  ch12= 2912428.723454   time= 0.123843   cunTime= 6.77181   edges= 28826   schur= 1   lambda= 538862000.602940   levenbergIter= 1
Iterations: 48  ch12= 2912428.663862   time= 0.167946   cunTime= 6.93976   edges= 28826   schur= 1   lambda= 2873930669.882344   levenbergIter= 3
Iterations: 49  ch12= 2912428.611790   time= 0.167937   cunTime= 7.1077   edges= 28826   schur= 1   lambda= 8082458057.531221   levenbergIter= 3
Iterations: 50  ch12= 2912428.604195   time= 0.190872   cunTime= 7.29942   edges= 28826   schur= 1   lambda= 195225771893.999390   levenbergIter= 4
Iterations: 51  ch12= 2912428.601486   time= 0.190274   cunTime= 7.48869   edges= 28826   schur= 1   lambda= 3951483133738.653320   levenbergIter= 4
Iterations: 52  ch12= 2912428.601486   time= 0.213657   cunTime= 7.70235   edges= 28826   schur= 1   lambda= 129482199326348192.000000   levenbergIter= 1
Draw poses: 7, points: 4118

```



1. 能否不要以 $[x, y, z]^T$ 的形式来参数化每个点？

答：逆深度参数化，6-D

$$P_w = (x_c \ y_c \ z_c \ \theta \ \phi \ \rho)$$

$$P_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} + \frac{1}{\rho} m(\theta, \phi)$$

其中， $[x_c, y_c, z_c]^T$ 为第一次观察到该 3D 路标点的相机的光心在世界坐标系下的位置， (θ, ϕ) 表示相机光轴到该 3D 路标点的旋转角度， ρ 为相机光心到该 3D 路标点的距离的倒数，另有：

$$m(\theta, \phi) = (\cos \phi \sin \theta, -\sin \phi, \cos \phi \cos \theta)^T$$

2. 取比 4×4 更大还是更小的 patch 好？

答：取小一点更好，因为不同视角下，即使看到同一个 3D 点，看到的周围的灰度值也不同，尤其对于一些边缘点，因此缩小 patch 会更容易匹配成功。

3. 直接法与特征点法在 BA 阶段有何不同？

答：直接法中 Jacobian 需要考虑灰度梯度，而特征点不用；直接法中需要考虑 patch 块，比如 4×4 ，即 16 个 error 项，而特征点法只考虑重投影误差，即 2 个 error 项。

4. Huber 的阈值如何设置？

答：假设误差项是高斯分布的，则误差项的平方服从卡方分布，然后确定误差项的自由度，以及置信度，根据自由度和置信度查找卡方分布表就能知道阈值是多少，一般置信度假设 0.95。

自由度对于 4×4 的 patch，其 error 维度为 16，因此自由度为 16。

卡方分布表 [\[编辑\]](#)

p-value = 1 - p_CDF.

χ^2 越大，p-value越小，则可信度越高。通常用p=0.05作为阈值，即95%的可信度。

常用的 χ^2 与p-value表如下：

自由度k \ P value (概率)	0.95	0.90	0.80	0.70	0.50	0.30	0.20	0.10	0.05	0.01	0.001
1	0.004	0.02	0.06	0.15	0.46	1.07	1.64	2.71	3.84	6.64	10.83
2	0.10	0.21	0.45	0.71	1.39	2.41	3.22	4.60	5.99	9.21	13.82
3	0.35	0.58	1.01	1.42	2.37	3.66	4.64	6.25	7.82	11.34	16.27
4	0.71	1.06	1.65	2.20	3.36	4.88	5.99	7.78	9.49	13.28	18.47
5	1.14	1.61	2.34	3.00	4.35	6.06	7.29	9.24	11.07	15.09	20.52
6	1.63	2.20	3.07	3.83	5.35	7.23	8.56	10.64	12.59	16.81	22.46
7	2.17	2.83	3.82	4.67	6.35	8.38	9.80	12.02	14.07	18.48	24.32
8	2.73	3.49	4.59	5.53	7.34	9.52	11.03	13.36	15.51	20.09	26.12
9	3.32	4.17	5.38	6.39	8.34	10.66	12.24	14.68	16.92	21.67	27.88
10	3.94	4.86	6.18	7.27	9.34	11.78	13.44	15.99	18.31	23.21	29.59